(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 463 409 A2**

(12) EUROPEAN PATENT APPLICATION

(21) Application number: **91109115.5**

(51) Int. Cl.5: **G10H 7/00**, G10H 1/04

(22) Date of filing: **04.06.91**

(30) Priority: **28.06.90 JP 171216/90**
28.06.90 JP 171217/90

(43) Date of publication of application:
02.01.92 Bulletin 92/01

(84) Designated Contracting States:
DE FR GB IT

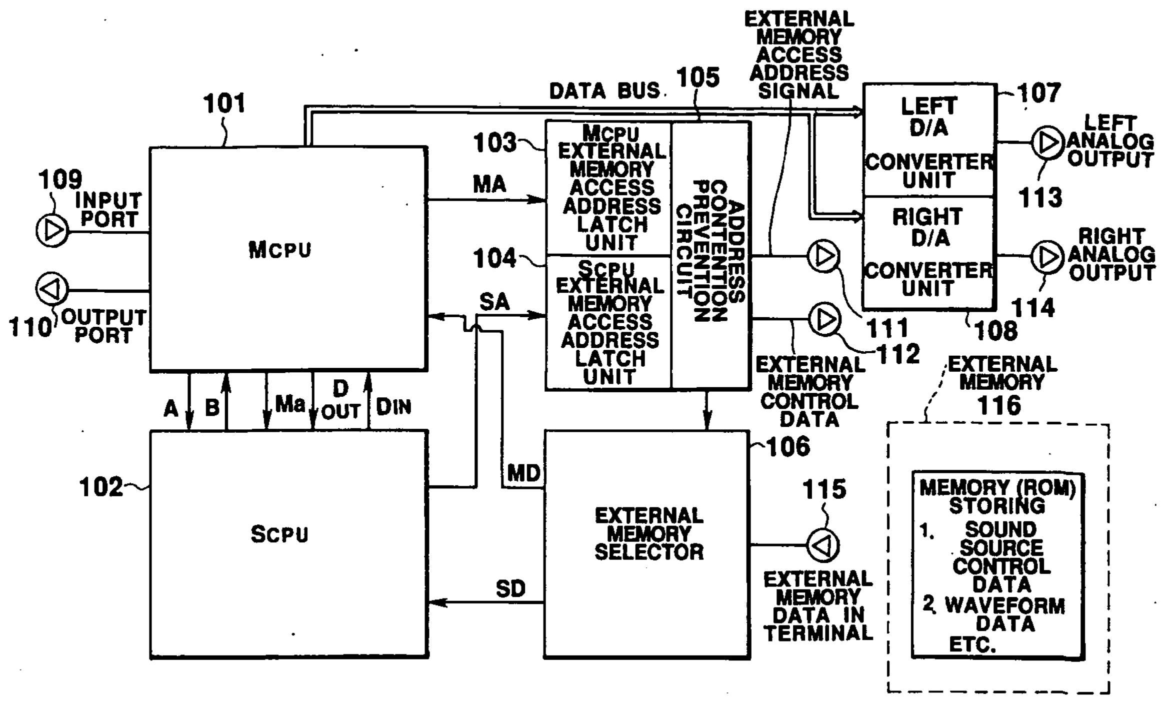
(71) Applicant: **CASIO COMPUTER COMPANY LIMITED**
6-1, 2-chome, Nishi-Shinjuku
Shinjuku-ku Tokyo(JP)

(72) Inventor: **Usami, Ryuji, c/o Patent Department., Dev. Div.**
CASIO COMPUTER CO., LTD., 3-2-1, Sakae-cho
Hamura-machi, Nishitama-gun, Tokyo190-11(JP)
Inventor: **Shiba, Kosuke, c/o Patent Department., Dev. Div.**
CASIO COMPUTER CO., LTD., 3-2-1, Sakae-cho
Hamura-machi, Nishitama-gun, Tokyo190-11(JP)
Inventor: **Daigo, Koichiro, c/o Patent Department., Dev. Div.**
CASIO COMPUTER CO., LTD., 3-2-1, Sakae-cho
Hamura-machi, Nishitama-gun, Tokyo190-11(JP)
Inventor: **Ogura, Kazuo, c/o Patent Department., Dev. Div.**
CASIO COMPUTER CO., LTD., 3-2-1, Sakae-cho
Hamura-machi, Nishitama-gun, Tokyo190-11(JP)
Inventor: **Hosoda, Jun, c/o Patent Department., Dev. Div.**
CASIO COMPUTER CO., LTD., 3-2-1, Sakae-cho
Hamura-machi, Nishitama-gun, Tokyo190-11(JP)

(74) Representative: **KUHNEN, WACKER & PARTNER**
Alois-Steinecker-Strasse 22 Postfach 1553
W-8050 Freising(DE)

(54) Musical tone waveform generation apparatus.

(57) A musical tone signal is generated by CPUs (101, 102) upon execution of a sound source processing program, associated with a modulation method stored in a memory (116). The generated musical tone signal is output at predetermined time intervals from D/A converters (107, 108). As for sound source processing based on the modulation method, at least one operator processing, and algorithm processing for determining an input/output relationship among the operator processing operations are independently executed.



DATA BUS
105
EXTERNAL MEMORY ACCESS ADDRESS SIGNAL
101
107
LEFT D/A CONVERTER UNIT
RIGHT D/A CONVERTER UNIT
LEFT ANALOG OUTPUT
RIGHT ANALOG OUTPUT
113
114
103
MCPU EXTERNAL MEMORY ACCESS ADDRESS LATCH UNIT
104
SCPU EXTERNAL MEMORY ACCESS ADDRESS LATCH UNIT
ADDRESS CONTENTION PREVENTION CIRCUIT
109
INPUT PORT
110
OUTPUT PORT
MCPU
MA
SA
111
112
108
EXTERNAL MEMORY CONTROL DATA
EXTERNAL MEMORY
116
A
B
Ma
D OUT
DIN
106
102
SCPU
MD
EXTERNAL MEMORY SELECTOR
115
EXTERNAL MEMORY DATA IN TERMINAL
SD
MEMORY (ROM) STORING
1. SOUND SOURCE CONTROL DATA
2. WAVEFORM DATA
ETC.

FIG.1

The present invention relates to a sound source processing method in a musical tone waveform generation apparatus.

Along with the development of the digital signal processing techniques and LSI processing techniques, various electronic musical instruments having good performance have been realized. In particular, the advent of an electronic musical instrument having a musical tone waveform generation apparatus for modulating an input signal generated based on performance data, and outputting it as a musical tone waveform is contributing to a great increase in music population regardless of professional and amateur musicians.

A musical tone waveform generation apparatus for an electronic musical instrument must perform large-volume, high-speed digital calculations. For this reason, a conventional musical tone waveform generation apparatus is constituted by a special-purpose sound source circuit which realizes an architecture equivalent to a musical tone generation algorithm based on a sound source method (for example, an FM method or a PCM method) by hardware components. The sound source method is realized by such a sound source circuit.

The above-mentioned sound source circuit has a large circuit scale regardless of the sound source method adopted. When the sound source circuit is realized by an LSI, it has a scale about twice that of a versatile data processing microprocessor because of fallowing four reasons. Firstly, the sound source circuit requires complicated address control for accessing waveform data on the basis of various performance data. Secondly, registers or the like for temporarily storing intermediate data obtained in the process of sound source generation processing must be arranged in the architecture corresponding to the sound source method. Thirdly, some modulation methods can variously change musical tone generation algorithms, and hardware arrangements corresponding to these methods are also required. Fourthly, in order to realize a polyphonic arrangement capable of simultaneously generating a plurality of musical tones, shift registers or the like for time-divisionally executing sound source processing in a hardware manner must be arranged everywhere.

As described above, since the conventional musical tone waveform generation apparatus is constituted by the special-purpose sound source circuit corresponding to the sound source method, its hardware scale is undesirably increased. This results in an increase in manufacturing cost in terms of, e.g., a yield in the manufacture of LSI chips, when the sound source circuit is realized by an LSI. This also results in an increase in size of the musical tone waveform generation apparatus.

When a sound source method is to be changed, or when the number of polyphonic channels is to be increased, the sound source circuit must be considerably modified, resulting in an increase in development cost.

When the conventional musical tone waveform generation apparatus is used as an electronic musical instrument, a control circuit, comprising, e.g., a microprocessor, for generating, based on performance data corresponding to a performance operation, data which can be processed by the sound source circuit, and for communicating performance data with another musical instrument, is required. The control circuit requires a sound source control program, corresponding to the sound source circuit, for supplying data corresponding to performance data to the sound source circuit as well as to a performance data processing program for processing performance data. In addition, these two programs must be synchronously operated. The development of such complicated programs causes a considerable increase in cost.

On the other hand, in recent years, high-performance microprocessors for performing versatile data processing have been developed, and a musical tone waveform generation apparatus for executing sound source processing in a software manner using such a microprocessor can be considered. However, no technique for synchronously operating a performance data processing program for processing performance data, and a sound source processing program for executing sound source processing on the basis of the performance data is available. In particular, since a processing time in the sound source processing program varies depending on the sound source method, a complicated timing control program for outputting generated musical tone data to a D/A converter is required. When the sound source processing is merely performed in a software manner, the processing programs are complicated very much, and processing of the high-speed sound source method such as a modulation method cannot be executed in terms of a processing speed and a program capacity. In particular, as described above, some modulation methods can variously change musical tone generation algorithms. If a microprocessor individually has sound source processing programs corresponding to such algorithms, this causes a demerit in terms of a program capacity, resulting in an expensive, large musical tone generation apparatus.

It is an object of the present invention to realize sound source processing based on a modulation method under the program control of a microprocessor without requiring a special-purpose sound source circuit.

It is another object of the present invention to realize sound source processing based on a modulation method, which can be operated in various musical tone generation algorithms under the program control of a microprocessor without requiring a special-purpose sound source circuit.

According to the first aspect of the present invention, there is provided a musical tone wave form generation apparatus comprising: storage means for storing a sound source processing program based on a predetermined modulation method; musical tone signal generation means for generating a musical tone signal on the basis of a process of the modulation method by executing the sound source processing program stored in the storage means; and musical tone signal output means for outputting the musical tone signal generated by the musical tone signal generation means at predetermined time intervals.

According to the musical tone waveform generation apparatus of the first aspect of the present invention, high-level sound source processing based on a modulation method can be realized without using a special-purpose sound source circuit, and since a constant output rate of a musical tone signal can be maintained upon operation of the musical tone signal output means, a musical tone waveform free from a distortion can be obtained.

According to the second aspect of the present invention, there is provided a musical tone waveform generation apparatus comprising: program storage means for storing a performance data processing program for processing performance data, and a sound source processing program, based on a modulation method, for obtaining a musical tone signal; address control means for controlling an address of the program storage means; data storage means for storing musical tone generation data necessary for generating a musical tone signal based on the modulation method; arithmetic processing means for performing arithmetic processing; program execution means for executing the performance data processing program and the sound source processing program stored in the program storage means while controlling the address control means, the data storage means, and the arithmetic processing means, the program execution means normally executing the performance data processing program to control musical tone generation data on the data storage means, executing the sound source processing program at predetermined time intervals, executing the performance data processing program again upon completion of the sound source processing program, and generating a musical tone signal by the modulation method on the basis of the musical tone generation data on the data storage means upon execution of the sound source processing program; and musical tone signal output means for holding the musical tone signal obtained when the program execution means executes the sound source processing program, and outputting the held musical tone signal at predetermined output time intervals.

According to the musical tone waveform generation apparatus of the second aspect of the present invention, the program storage means, the address control means, the data storage means, the arithmetic processing means, and the program execution means have the same arrangement as a versatile microprocessor, and no special-purpose sound source circuit is required at all. The musical tone signal output means is versatile in the category of a musical tone waveform generation apparatus although it has an arrangement different from that of a versatile microprocessor.

The circuit scale of the overall musical tone waveform generation apparatus can be greatly reduced, and when the apparatus is realized by an LSI, the same manufacturing technique as that of a normal processor can be adopted. Since the yield of chips can be increased, manufacturing cost can be greatly reduced. Since the musical tone signal output means can be constituted by simple latch circuits, addition of this circuit portion causes almost no increase in manufacturing cost.

When a modulation method is required to be switched between, e.g., a phase modulation method and a frequency modulation method, or when the number of polyphonic channels is required to be changed, a sound source processing program stored in the program storage means need only be changed to meet the above requirements. Therefore, the development cost of a new musical tone waveform generation apparatus can be greatly reduced, and a new modulation method can be presented to a user by means of, e.g., a ROM card.

The above-mentioned effects can be provided since the second aspect of the present invention can realize the following program and data architectures.

More specifically, the second aspect of the present invention uses the data architecture for storing musical tone generation data necessary for generating musical tones in a modulation method on the data storage means. When a performance data processing program is executed, the musical tone generation data on the data storage means are controlled, and when a sound source processing program is executed, musical tone signals are generated on the basis of the musical tone generation data on the data storage means. A data communication between the performance data processing program and the sound source processing program is performed via musical tone generation data on the data storage means, and access of one program to the data storage means can be performed regardless of an execution state of the other program. Therefore, the two programs can have substantially independent module arrangements, and hence, a simple and efficient program architecture can be attained.

In addition to the data architecture, the second aspect of the present invention uses the following program architecture. That is, the performance data processing program is normally executed for scanning of keyboard keys and various setting switches, demonstration performance control, and the like. During execution of this program, the sound source processing program is executed at predetermined time intervals, and upon completion of the processing, the control returns to the performance data processing program. Thus, the sound source processing program forcibly interrupts the performance data processing program on the basis of an interrupt signal generated from the interrupt control means at predetermined time intervals. For this reason, the performance data processing program and the sound source processing program need not be synchronized.

When the program execution means executes the sound source processing program, its processing time changes depending on the type of modulation method or a selected musical tone generation algorithm in the modulation method. However, the change in processing time can be absorbed by the musical tone signal output means. Therefore, no complicated timing control program for outputting musical tone signals to, e.g., a D/A converter is required.

As described above, the data architecture for attaining a data link between the performance data processing program and the sound source processing program via musical tone generation data on the data storage means, and the program architecture for executing the sound source processing program at predetermined time intervals while interrupting the performance data processing program are realized, and the musical tone signal output means is arranged. Therefore, sound source processing under the efficient program control can be realized by substantially the same arrangement as a versatile processor.

According to the third aspect of the present invention, there is provided a musical tone waveform generation apparatus comprising: storage means for storing a sound source processing program associated with a modulation method, having an operator processing program for executing operator processings, and an algorithm processing program for executing algorithm processing for determining an input/output relationship among operator processing; musical tone signal generation means for generating a musical tone signal by executing the operator processing operations based on the operator processing program at a time, and executing the algorithm processing at a time based on the algorithm processing program independently of the operator processing program; and musical tone signal output means for outputting the musical tone signal generated by the musical tone signal generation means at predetermined output time intervals.

According to the musical tone waveform generation apparatus of the third aspect of the present invention, high-level sound source processing which can be operated in various musical tone generation algorithms can be realized without using a special-pourpose sound source circuit, and a constant output rate of a musical tone signal can be maintained upon operation of the musical tone signal output means. Therefore, a musical tone waveform free from a distortion can be obtained.

According to the fourth aspect of the present invention, there is provided a musical tone waveform generation apparatus comprising: program storage means for storing a performance data processing program for processing performance data, and a sound source processing program based on a modulation method for obtaining a musical tone signal, the sound source processing program having a processing architecture in which algorithm processing operations for determining an input/output relationship among a plurality of operator processing operations are executed at a time after or before execution of the plurality of operator processing operations at a time as modulation processing units; address control means for controlling an address of the program storage means; data storage means for storing musical tone generation data necessary for generating a musical tone signal based on the modulation method; arithmetic processing means for processing data; program execution means for executing the performance data processing program and the sound source processing program stored in the program storage means while controlling the address control means, the data storage means, and the arithmetic processing means, for normally executing the performance data processing program to control musical tone generation data on the data storage means, for executing the sound source processing program at predetermined time intervals, for executing the performance data processing program again upon completion of the sound source processing program, and for generating a musical tone signal by the modulation method on the basis of the musical tone generation data on the data storage means upon execution of the sound source processing program; and musical tone signal output means for holding the musical tone signal obtained when the program execution means executes the sound source processing program, and outputting the held musical tone signal at predetermined output time intervals.

The musical tone waveform generation apparatus according to the fourth aspect of the present invention has, as an architecture of the sound source processing program, a processing architecture for simultaneously execting algorithm processing operations for determining the I/O (input/output) relationship of operator processing operations before or after simultaneous execution of the operator processing operations as modulation processing units. Since a conventional apparatus has a processing architecture in that the I/O relationship of the next operator is determined by a designated algorithm upon completion of one operator processing, a plurality of types of sound source processing programs including operator processing portions must be prepared in units of algorithms. In contrast to this, in the musical tone waveform generation apparatus according to the fourth aspect of the present invention, a plurality of types of only algorithm processing portions are prepared, and are switched as needed even when sound source processing is to be performed by an algorithm selected from a plurality of algorithms. Therefore, the sound source processing program can be rendered very compact.

This invention can be more fully understood from the following detailed description when taken in conjunction with the accompanying drawings, in which:

Fig. 1 is a block diagram showing the overall arrangement according to an embodiment of the present invention;

Fig. 2 is a block diagram showing an internal of a master CPU;

Fig. 3 is a block diagram showing an internal arrangement of a slave CPU;

Figs. 4A to 4D are flow charts showing operations of the overall arrangement of this embodiment;

Fig. 5 is a schematic view showing the relationship among the main operation flow charts and interrupt pro

cessing; Fig. 6A is a diagram of a conventional D/A converter unit;

Fig. 6B is a diagram of a D/A converter unit according to this embodiment;

Fig. 7 is a timing chart in D/A conversion;

Fig. 8A illustrates an arrangement of a function key and a keyboard key;

Fig. 8B is an explanatory view of keyboard keys;

Fig. 9 shows storage areas in units of tone generation channels on a RAM;

Fig. 10 is a schematic diagram upon selection of a sound source processing method of each tone generation channel;

Fig. 11 shows an architecture of data formats in units of sound source methods on the RAM;

Fig. 12 shows buffer areas on the RAM;

Fig. 13 is an operation flow chart of sound source processing based on a PCM method;

Fig. 14 is an operation flow chart of sound source processing based on a DPCM method;

Figs. 15A and 15B are graphs for explaining the principle when an interpolation value X_Q is obtained using a difference D and the present address A_F;

Fig. 16A is an operation flow chart of sound source processing based on an FM method (Part 1);

Fig. 16B is an operation flow chart of the sound processing based on the FM method (Part 1);

Fig. 17A is an operation flow chart of sound source processing based on a TM method (Part 1);

Fig. 17B is a chart showing an algorithm of the source processing based on the TM method (Part 1);

Figs. 18A to 18D are charts showing algorithms in a modulation method;

Fig. 19 is an operation flow chart of sound source processing based on an FM method (Part 2);

Fig. 20 is an operation flow chart of sound source processing based on a TM method (Part 2);

Fig. 21 is an operation flow chart of a first modification of the modulation method;

Fig. 22A is an operation flow chart of operator 1 processing based on the FM method according to the first modification;

Fig. 22B is a chart showing an arithmetic algorithm per operator in the operator 1 processing based on the FM method according to the first modification;

Fig. 23A is an operation flow chart of operator 1 processing based on the TM method according to the first modification;

Fig. 23B is a chart showing an arithmetic algorithm per operator in the operator 1 processing based on the TM method according to the first modification;

Fig. 24 is an operation flow chart of algorithm processing according to the first modification;

Fig. 25 is an operation flow chart of a second modification of the modulation method;

Fig. 26 is an operation flow chart of algorithm processing according to the second modification;

Fig. 27 shows an arrangement of some function keys;

Figs. 28A and 28B show examples of assignments of sound source methods to tone generation channels;

Fig. 29 is an operation flow chart of function key processing;

Fig. 30 is an operation flow chart of the first embodiment of ON event keyboard key processing;

Fig. 31 is an operation flow chart of the second embodiment of ON event keyboard key processing; and

Fig. 32 is an operation flow chart of an embodiment of OFF event keyboard key processing.

Summary of Embodiment

The summary of this embodiment will be described below.

Fig. 1 is a block diagram showing the overall arrangement of this embodiment. In Fig. 1, components other than an external memory 116 are constituted in one chip. Of these components, two, i.e., master and slave CPUs (central processing units) exchange data to share sound source processing for generating musical tones.

In, e.g., a 16-channel polyphonic system, 8 channels are processed by a master CPU 101, and the remaining 8 channels are processed by a slave CPU 102.

The sound source processing is executed in a software manner, and sound source methods such as PCM (Pulse Code Moduration) and DPCM (Differential PCM) methods, and sound source methods based on modulation methods such as FM and phase modulation methods are assigned in units of tone generation channels.

A sound source method is automatically designated for tone colors of specific instruments, e.g., a trumpet, a tuba, and the like. For tone colors of other instruments, a sound source method can be selected by a selection switch, and/or can be automatically selected in accordance with a performance tone range, a performance strength such as a key touch, and the like.

In addition, different sound source methods can be assigned to two channels for one ON event of a key. That is, for example, the PCM method can be assigned to an attack portion, and the FM method can be assigned to a sustain portion.

Furthermore, in, e.g., the FM method, when software processing is executed by a versatile CPU according to a sound source processing algorithm, it requires too much time. However, this embodiment can also solve this problem.

Arrangement of This Embodiment

The embodiment of the present invention will be described below with reference to the accompanying drawings.

In Fig. 1, the external memory 116 stores musical tone control parameters such as target values of envelope values, a musical tone waveform in the PCM (pulse code modulation) method, a musical tone differential waveform in the DPCM (differential PCM) method, and the like.

The master CPU (to be abbreviated to as an MCPU hereinafter) 101 and the slave CPU (to be abbreviated to as an SCPU hereinafter) 102 access the data on the external memory 116 to execute sound source processing while sharing processing operations. Since these CPUs 101 and 102 commonly use waveform data of the external memory 116, a contention may occur when data is loaded from the external memory 116. In order to prevent this contention, the MCPU 101 and the SCPU 102 output an address signal for accessing the external memory, and external memory control data from output terminals 111 and 112 of an access address contention prevention circuit 105 via an external memory access address latch unit 103 for the MCPU, and an external memory access address latch unit 104 for the SCPU. Thus, a contention between addresses from the MCPU 101 and the SCPU 102 can be prevented.

Data read out from the external memory 116 on the basis of the designated address is input from an external memory data input terminal 115 to an external memory selector 106. The external memory selector 106 separates the readout data into data to be input to the MCPU 101 via a data bus MD and data to be input to the SCPU 102 via a data bus SD on the basis of a control signal from the address contention prevention circuit 105, and inputs the separated data to the MCPU 101 and the SCPU 102. Thus, a contention between readout data can also be prevented.

After the MCPU 101 and the SCPU 102 perform corresponding sound source processing operations of the input data by software, musical tone data of all the tone generation channels are accumulated, and a left-channel analog output and a right-channel analog output are then output from a left output terminal 113 of a left D/A converter unit 107 and a right output terminal 114 of a right D/A converter unit 108, respectively.

Fig. 2 is a block diagram showing an internal arrangement of the MCPU 101.

In Fig. 2, a control ROM 201 stores a musical tone control program (to be described later), and sequentially outputs program words (commands) addressed by a ROM address controller 205 via a ROM address decoder 202. This embodiment employs a next address method. More specifically, the word length of each program word is, e.g., 28 bits, and a portion of a program word is input to the ROM address controller 205 as a lower bit portion (intra-page address) of an address to be read out next. Note that the SCPU 101 may comprise a conventional program counter type CPU insted of control ROM 201.

A command analyzer 207 analyzes operation codes of commands output from the control ROM 201, and sends control signals to the respective units of the circuit so as to execute designated operations.

When an operand of a command from the control ROM 201 designates a register, the RAM address controller 204 designates an address of a corresponding internal register of a RAM 206. The RAM 206 stores various musical tone control data (to be described later with reference to Figs. 11 and 12) for eight tone generation channels, and includes various buffers (to be described later) or the like. The RAM 206 is used in sound source processing (to be described later).

When a command from the control ROM 201 is an arithmetic command, an ALU unit 208 and a multiplier 209 respectively execute an addition/subtraction, and a multiplication on the basis of an instruction from the command analyzer 207.

On the basis of an internal hardware timer (not shown), an interrupt controller 203 supplies a reset cancel signal A to the SCPU 201 (Fig. 1) and an interrupt signal to the D/A converter units 107 and 108 (Fig. 1) at predetermined time intervals.

In addition to the above-mentioned arrangement, the MCPU 101 shown in Fig. 2 comprises the following interfaces associated with various buses: an interface 215 for an address bus MA for addressing the external memory 116 to access it; an interface 216 for the data bus MD for exchanging the accessed data with the MCPU 101 via the external memory selector 106; an interface 212 for a bus Ma for addressing the internal RAM of the SCPU 102 so as to execute data exchange with the SCPU 102; an interface 213 for a data bus D_{OUT} used by the MCPU 101 to write data in the SCPU 102; an interface 214 for a data bus D_{IN} used by the MCPU 101 to read data from the SCPU 102; an interface 217 for a D/A data transfer bus for transferring final output waveforms to the left and right D/A converter units 107 and 108; and input and output ports 210 and 211 for exchanging data with an external switch unit or a keyboard unit (Fig. 8).

Fig. 3 shows the internal arrangement of the SCPU 102.

Since the SCPU 102 executes sound source processing upon reception of a processing start signal from the MCPU 101, it does not comprise an interrupt controller corresponding to the controller 203 (Fig. 2), I/O ports, corresponding to the ports 210 and 211 (Fig. 2), for exchanging data with an external circuit, and an interface, corresponding to the interface 217 (Fig. 2) for outputting musical tone signals to the left and right D/A converter units 107 and 108. Other circuits 301, 302, and 304 to 309 have the same functions as those of the circuits 201, 202, and 204 to 209 shown in Fig. 2. Interfaces 303, and 310 to 313 are arranged in correspondence with the interface 212 to 216 shown in Fig. 2. Note that the internal RAM address of the SCPU 102 designated by the MCPU 101 is input to the RAM address controller 304. The RAM address controller 304 designates an address of the RAM 306. Thus, accumulated waveform data for eight tone generation channels generated by the SCPU 102 and held in the RAM 306 are output to the MCPU 101 via the data bus D_{IN}. This will be described later.

In addition to the above-mentioned arrangement, in this embodiment, function keys 801, keyboard keys 802, and the like shown in Fig. 8 are connected to the input port 210 of the MCPU 101. These portions substantially constitute an instrument operation unit.

The D/A converter unit as one characteristic feature of the present invention will be described below.

Fig. 6B shows the internal arrangement of the left or right D/A converter unit 107 or 108 (the two converter units have the same contents) shown in Fig. 1. One sample data of a musical tone generated by sound source processing is input to a latch 601 via a data bus. When the clock input terminal of the latch 601 receives a sound source processing end signal from the command analyzer 207 (Fig. 2) of the MCPU 101, musical tone data for one sample on the data bus is latched by the latch 601, as shown in Fig. 7.

A time required for the sound source processing changes depending on the sound source processing software program. For this reason, a timing at which each sound source processing is ended, and musical tone data is latched by the latch 601 is not fixed. For this reason, as shown in Fig. 6A, an output from the latch 601 cannot be directly input to a D/A converter 603.

In this embodiment, as shown in Fig. 6B, the output from the latch 601 is latched by a latch 602 in response to an interrupt signal equal to a sampling clock interval output from the interrupt controller 203, and is output to the D/A converter 603 at predetermined time intervals.

Since a change in processing time can be absorbed using the two latches 601 and 602, no complicated control program for outputting musical tone data to a D/A converter 603 is required.

Overall Operation of This Embodiment

The overall operation of this embodiment will be described below.

In this embodiment, basically, the MCPU 101 is mainly operated, and repetitively executes a series of processing operations in steps S402 to S410, as shown in the main flow chart of Fig. 4A. The sound source processing is performed by interrupt processing. More specifically, the MCPU 101 and the SCPU 102 are interrupted at predetermined time intervals, and each CPU executes sound source processing for generating musical tones for eight channels. Upon completion of this processing, musical tone waveforms for 16 channels are added, and are output from the left and right D/A converter units 107 and 108. Thereafter, the control returns from the interrupt state to the main flow. Note that the above-mentioned interrupt processing is periodically executed on the basis of the internal hardware timer in the interrupt controller 203 (Fig. 2). This period is equal to a sampling period when a musical tone is output.

The schematic operation of this embodiment has been described. The operation of this embodiment will be described in detail below with reference to Figs. 4A to 4D.

When the interrupt controller 203 interrupts repetitively executed processing operations in steps S402 to S410 in the main flow chart of Fig. 4A, MCPU interrupt processing shown in Fig. 4B and SCPU interrupt processing shown in Fig. 4C are simultaneously started. "Sound source processing" in Figs. 4B and 4C is shown in Fig. 4D.

The main flow chart of Fig. 4A shows a processing flow executed by the MCPU 101 in a state wherein no interrupt signal is supplied from the interrupt controller 203.

When the power switch is turned on, the system e.g., the contents of the RAM 206 in the MCPU 101 are initialized (S401).

The function keys externally connected to the MCPU 101, e.g., tone color switches, and the like (Fig. 27), are scanned (S402) to fetch respective switch states from the input port 210 to a key buffer area in the RAM 206. As a result of scanning, a function key whose state is changed is discriminated, and processing of a corresponding function is executed (S403). For example, a musical tone number or an envelope number is set, or if optional functions include a rhythm performance function, a rhythm number is set.

Thereafter, states of ON keyboard keys are fetched in the same manner as the function keys (S404), and keys whose states are changed are discriminated, thus executing key assignment processing (S405).

When a demonstration performance key of the function keys 801 (Fig. 8) is depressed, demonstration performance data (sequencer data) are sequentially read out from the external memory 116 to execute, e.g., key assignment processing (S406). When a rhythm start key is depressed, rhythm data are sequentially read out from the external memory 116 to execute, e.g., key assignment processing (S407).

Thereafter, timer processing is executed (S408). More specifically, time data which is incremented by interrupt timer processing (S412) (to be described later) is compared with time control sequencer data sequentially read out for demonstration performance control or time control rhythm data read out for rhythm performance control, thereby executing time control when a demonstration performance in step S406 or a rhythm performance in step S407 is performed.

In tone generation processing in step S409, pitch envelope processing, and the like are executed. In this processing, an envelope is added to a pitch of a musical tone to be generated, and pitch data is set in a corresponding tone generation channel.

Furthermore, one flow cycle preparation processing is executed (S410). In this processing, processing for changing a state of a tone generation channel assigned with a note number corresponding to an ON event detected in the keyboard key processing in step S405 to an "ON event" state, and processing for changing a state of a tone generation channel assigned with a note number corresponding to an OFF event to a "muting" state, and the like are executed.

The MCPU interrupt processing shown in Fig. 4B will be described below.

When the interrupt controller 203 of the MCPU 101 interrupts the MCPU 101, the processing in the main flow chart shown in Fig. 4A is interrupted, and the MCPU interrupt processing in Fig. 4B is started. In this case, control is made to avoid contents of registers to be subjected to write access in the main flow program in Fig. 4A from being rewritten in the MCPU interrupt processing program. For this reason, the MCPU interrupt processing uses registers different from those used in the main flow program. As a result, register save/restoration processing normally executed at the beginning and end of interrupt processing can be omitted. Thus, transition between the processing of the main flow chart shown in Fig. 4A and the MCPU interrupt processing can be quickly performed.

Subsequently, in the MCPU interrupt processing, sound source processing is started (S411). The sound source processing is shown in Fig. 4D.

Simultaneously with the above-mentioned operations, the interrupt controller 203 of the MCPU 101 outputs the SCPU reset cancel signal $\overline{A}$ (Fig. 1) to the ROM address controller 305 of the SCPU 102, and the SCPU 102 starts execution of the SCPU interrupt processing (Fig. 4C).

Sound source processing (S415) is started in the SCPU interrupt processing almost simultaneously with the source processing (S411) in the

MCPU interrupt processing. In this manner, since each of the MCPU 101 and the SCPU 102 simultaneously executes sound source processing of eight tone generation channels, the sound source processing for 16 tone generation channels can be executed in a processing time for eight tone generation channels, and a processing speed can be almost doubled (the interrupt processing will be described later with reference to Fig. 5).

In the interrupt timer processing in step S412, the value of time data (not shown) on the RAM 206 (Fig. 2) is incremented by utilizing the fact that the interrupt processing shown in Fig. 4B is executed for every predetermined sampling period. More specifically, a time elapsed from power-on can be detected based on the value of the time data. The time data obtained in this manner is used in time control in the timer processing in step S408 in the main flow chart shown in Fig. 4A.

The MCPU 101 then waits for an SCPU interrup processing end signal B from the SCUP 102 after interrupt timer processing in step S412 (S413).

Upon completion of the sound source processing in step S415 in Fig. 4C, the command analyzer 307 of the SCPU 102 supplies an SCPU processing end signal B (Fig. 1) to the ROM address controller 205 of the MCPU 101. In this manner, YES is determined in step S413 in the MCPU interrupt processing in Fig. 4B.

As a result, waveform data generated by the SCPU 102 are written in the RAM 206 of the MCPU 101 via the data bus D_{IN} shown in Fig. 1 (S414). The waveform data are stored in a predetermined buffer area (a buffer B to be described later) on the RAM 306 of the SCPU 102. The command analyzer 207 of the MCPU 101 designates addresses of the buffer area to the RAM address controller 304, thus reading the waveform data.

In step S414', the contents of the buffer area B are latched by the latches 601 (Fig. 6) of the left and right D/A converter units 107 and 108.

The operation of the sound source processing executed in step S411 in the MCPU interrupt processing or in step S415 in the SCPU interrupt processing will be described below with reference to the flow chart of Fig. 4D.

A waveform addition area on the RAM 206 or 306 is cleared (S416). Then, sound source processing is executed in units of tone generation channels (S417 to S424). After the sound source processing for the eighth channel is completed, waveform data obtained by adding those for eight channels is obtained in the buffer area B. These processing operations will be described in detail later.

Fig. 5 is a schematic flow chart showing the relationship among the processing operations of the flow charts shown in Figs. 4A, 4B, and 4C. As can be seen from Fig. 5, the MCPU 101 and the SCPU 102 share the sound source processing.

Given processing A (the same applies to B, C,..., F) is executed (S501). This "processing" corresponds to, for example, "function key processing", or "keyboard key processing" in the main flow chart shown in Fig. 4A. Thereafter, the MCPU interrupt processing and the SCPU interrupt processing are executed, so that the MCPU 101 and the SCPU 102 simultaneously start sound source processing (S502 and S503). Upon completion of the SCPU interrupt processing of the SCPU 102, the SCPU processing end signal B is input to the MCPU 101. In the MCPU interrupt processing, the sound source processing is ended earlier than the SCPU interrupt processing, and the MCPU waits for the end of the SCPU interrupt processing the SCPU processing end signal B is discriminated in the MCPU interrupt processing, waveform data generated by the SCPU 102 is supplied to the MCPU 101, and is added to the waveform data generated by the MCPU 101. The waveform data is then output to the left and right D/A converter units 107 and 108. Thereafter, the control returns to some processing B in the main flow chart.

The above-mentioned operations are repeated (S504 to S516) while executing the sound source processing for all the tone generation channels (16 channels as a total of those of the MCPU 101 and the SCPU 102). The repetition processing continues as long as musical tones are being produced.

Data Architecture in Sound Source Processing

The sound source processing executed in step S411 (Fig. 4B) and step S415 (Fig. 4C) will be described in detail below.

In this embodiment, as described above, the two CPUs, i.e., the MCPU 101 and the SCPU 102 share the sound source processing in units of eight channels. Data for the sound source processing for eight channels are set in areas corresponding to the respective tone generation channels in the RAMs 206 and 306 of the MCPU 101 and the SCPU 102, as shown in Fig. 9.

Buffers BF, BT, B, and M are allocated on the RAM, as shown in Fig. 12.

In each tone generation channel area shown in Fig. 9, an arbitrary sound source method can be set by an operation (to be described in detail later), as schematically shown in Fig. 10. When the sound source method is set, data are set in each tone generation channel area in Fig. 9 in a data format of the corresponding sound source method, as shown in Fig. 11. In this embodiment, as will be described later, different sound methods can be assigned to the tone generation channels.

In Table 1 showing the data formats of the respective sound source methods shown in Fig. 11, G indicates a sound source method number for identifying the sound source methods. A represents an address designated when waveform data is read out in the sound source processing, and A_I, A_1, and A_2 represent integral parts of current addresses, and directly correspond to addresses of the external memory 116 (Fig. 1) where waveform data are stored. A_F represents a decimal part of the current address, and is used for interpolating waveform data read out from the external memory 116.

A_E and A_L respectively represent end and loop addresses. P_I, P_1 and P_2 represent integral parts of pitch data, and P_F represents a decimal part of pitch data. For example, $P_I = 1$ and $P_F = 0$ express a pitch of an original tone, PI = 2 and PF = 0 express a pitch higher than the original pitch by one octave, and $P_I = 0$ and $P_F = 0.5$ express a pitch lower by one octave.

X_P represents previous sample data, and X_N represents the next sample data. D represents a difference between two adjacent sample data, and E represents an envelope value. Furthermore, O represents an output value, and C represents a flag which is used when a sound source method to be assigned to a tone generation channel is changed in accordance with performance data, as will be described later.

Various other control data will be described in descriptions of the respective sound source methods.

When data shown in Fig. 11 are stored in the RAMs 206 and 306 of the MCPU 101 and the SCPU 102, and the sound source methods (to be described later) are determined, data are set in units of channels shown in Fig. 9 in the format shown in Fig. 11.

The sound source processing operations of the respective sound source methods executed using the above-mentioned data architecture will be described below in turn. These sound source processing operations are realized by analyzing and executing a sound source processing program stored in the control ROM 201 or 301 by the command analyzer 207 or 307 of the MCPU 101 or the SCPU 102. Assume that the processing is executed under this condition unless otherwise specified.

In the flow chart shown in Fig. 4D, in the sound source processing (one of steps S417 to S424) for each channel, the sound source method No. data G of the data in the data format (Table 1) shown in Fig. 11 stored in the corresponding tone generation channel of the RAM 206 or 306 is discriminated to determine sound source processing of a sound source method to be described below.

Sound Source Processing Based on PCM Method

When the sound source method No. data G indicates the PCM method, sound source processing based on the PCM method shown in the operation flow chart of Fig. 13 is executed. Variables in the flow chart are data in a PCM format of Table 1 shown in Fig. 11, which data are stored in the corresponding tone generation channel area (Fig. 9) of the RAM 206 or 306 of the MCPU 101 or the SCPU 102.

Of an address group of the external memory 116 (Fig. 1) where PCM waveform data are stored, an address where waveform data as an object to be currently processed is stored is assumed to be (A_I, A_F) shown in Fig. 15A.

Pitch data (P_I, P_F) is added to the current address (S1301). The pitch data corresponds to the type of an ON key of the keyboard keys 801 shown in Fig. 8.

It is then checked if the integral part A_I of the sum address is changed (S1302). If NO in step S1302, an interpolation data value O corresponding to the decimal part A_F of the address (Fig. 15A) is calculated by arithmetic processing $D \times A_F$ using a difference D as a difference between sample data X_N and X_P at addresses (A_I+1) and A_I (S1307). Note that the difference D has already been obtained by the sound source processing at previous interrupt timing (see step S1306 to be described later).

The sample data X_P corresponding to the integral part A_I of the address is added to the interpolation data value O to obtain a new sample data value O (corresponding to X_Q in Fig. 15A) corresponding to the current address (A_I, A_F) (S1308).

Thereafter, the sample data is multiplied with the envelope value E (S1309), and the content of the obtained data O is added to a value held in the waveform data buffer B (Fig. 12) in the RAM 206 or 306 of the MCPU 101 or the SCPU 102 (S1310).

Thereafter, the control returns to the main flow chart shown in Fig. 4A. The control is interrupted in the next sampling period, and the operation flow chart of the sound source processing shown in Fig. 13 is executed again. Thus, pitch data (P_I, P_F) is added to the present address (A_I, A_F) (S1301).

The above-mentioned operations are repeated until the integral part A_I of the address is changed (S1302).

Before the integral part is changed, the sample data X_P and the difference D are left unchanged, and only the interpolation data O is updated in accordance with the address A_F. Thus, every time the address A_F is updated, new sample data X_Q is obtained.

If the integral part A_I of the current address is changed (S1302) as a result of addition of the current address (A_I, A_F) and the pitch data (P_I, P_F) in step S1301, it is checked if the address A_I has reached or exceeded the end address A_E (S1303).

If YES in step S1303, the next loop processing is executed. More specifically, a value (A_I - A_E) as a difference between the updated current address A_T and the end address A_E is added to the loop address A_L to obtain a new current address (A_I, A_F). A loop reproduction is started from the obtained new current address A_I (S1304). The end address A_E is an end address of an area of the external memory 116 (Fig. 1) where PCM waveform data are stored. The loop address A_L is an address of a position where a player wants to repeat an output of a waveform, and known loop processing is realized by the PCM method.

If NO in step S1303, the processing in step S1304 is not executed.

Sample data is then updated. In this case, sample data corresponding to the new updated current address A_T and the immediately preceding address (A_I-1) are read out as X_N and X_P from the external memory 116 (Fig. 1) (S1305).

Furthermore, the difference so far is updated with a difference D between the updated data X_N and X_P (S1306).

The following operation is as described above.

In this manner, waveform data by the PCM method for one channel is generated.

Sound Source Processing Based on DPCM Method

The sound source processing based on the DPCM method will be described below.

The operation principle of the DPCM method will be briefly described below with reference to Fig. 15B.

In Fig. 15B, sample data X_P corresponding to an address A_I of the external memory 116 (Fig. 1) is obtained by adding sample data corresponding to an address (A_I-1) (not shown) to a difference between the sample data corresponding to the address (A_I-1) and sample data corresponding to the address A_I.

A difference D with the next sample data is written at the address A_I of the external memory 116 (Fig. 1). Sample data at the next address (A_I+1) is obtained by X_P + D.

In this case, if the decimal part of the current address is represented by A_F, as shown in Fig. 15B, sample data corresponding to the current address A_F is obtained by $X_P + D \times A_F$.

In this manner, in the DPCM method, a difference D between sample data corresponding to the current address and the next address is read out from the external memory 116 (Fig. 1), and is added to the current sample data to obtain the next sample data, thereby sequentially forming waveform data.

If the DPCM method is adopted, when a waveform such as a voice or a musical tone which generally has a small difference between adjacent samples is to be quantized, quantization can be performed by a smaller number of bits as compared to the normal PCM method.

The operation of the above-mentioned DPCM method will be described below with reference to the operation flow chart shown in Fig. 14. Variables in the flow chart are DPCM data in Table 1 shown in Fig. 11, which data are stored in the corresponding tone generation channel area (Fig. 9) on the RAM 206 or 306 of the MCPU 101 or the SCPU 102.

Of addresses on the external memory 116 (Fig. 1) where DPCM differential waveform data are stored, an address where waveform data as an object to be currently processed is stored is assumed to be (A_I, A_F) shown in Fig. 15B.

Pitch data (P_I, P_F) is added to the current address (A_I, A_F) (S1401).

It is then checked if the integral part A_I of the sum address is changed (S1402). If NO in step S1402, an interpolation data value $\underline{O}$ corresponding to the decimal part A_F of the address is calculated by arithmetic processing $D \times A_F$ using a difference D at the address A_I in Fig. 15B (S1414). Note that the difference D has already been obtained by the sound source processing at the previous interrupt timing (see steps S1406 and S1410 to be described later).

The interpolation data value $\underline{O}$ is added to sample data X_P corresponding to the integral part A_I of the address to obtain a new sample data value $\underline{O}$ (corresponding X_Q in Fig. 15B) corresponding to the current address (A_I, A_F) (S1415).

Thereafter, the sample data value $\underline{O}$ is multiplied with an envelope value E (S1416), and the obtained value is added to a value stored in the waveform data buffer $\underline{B}$ (Fig. 12) in the RAM 206 or 306 of the MCPU 101 or the SCPU 102 (S1417).

Thereafter, the control returns to the flow chart shown in Fig. 4A. The control is interrupted in the next sampling period, and the operation flow chart of the sound source processing shown in Fig. 14 is executed again. Thus, pitch data (P_I, P_F) is added to the current address (A_I, A_F) (S1401).

The above-mentioned operations are repeated until the integral part A_I of the address is changed.

Before the integral part is changed, the sample data X_P and the difference D are left unchanged, and only the interpolation data $\underline{O}$ is updated in accordance with the address A_F. Thus, every time the address A_F is updated, new sample data X_Q is obtained.

If the integral part A_I of the present address is changed (S1402) as a result of addition of the current address (A_I, A_F) and the pitch data (P_I, P_F) in step S1401, it is checked if the address A_I has reached or exceeded the end address A_E (S1403).

If NO in step S1403, sample data corresponding to the integral part A_I of the updated current address is calculated by the loop processing in steps S1404 to S1407. More specifically, a value before the integral part A_I of the current address is changed is stored in a variable "old A_I" (see the column of DPCM in Table 1 shown in Fig. 11). This can be realized by repeating processing in step S1406 or S1413 (to be described later). The old A_I value is sequentially incremented in S1406, and differential waveform data in the external memory 116 (Fig. 1) addressed by the old A_I values are read out as D in step S1407. The readout data D are sequentially accumulated on sample data X_P in step S1405. when the old A_I value becomes equal to the integral part A_I of the changed current address, the sample data X_P has a value corresponding to the integral part A_I of the changed current address.

When the sample data X_P corresponding to the integral part A_I of the current address is obtained in this manner, YES is determined in step S1404, and the control starts the arithmetic processing of the interpolation value (S1414) described above.

The above-mentioned sound source processing is repeated at the respective interrupt timings, and when the judgment in step S1403 is changed to YES, the control enters the next loop processing.

An address value (A_I-A_E) exceeding the end address A_E is added to the loop address A_L, and the obtained address is defined as an integral part A_I of a new current address (S1408).

An operation for accumulating the difference D several times depending on an advance in address from the loop address A_L is repeated to calculate sample data X_P corresponding to the integral part A_I of the new current address. More specifically, sample data X_P is initially set as the value of sample data X_{PL} (see the column of DPCM in Table 1 shown in Fig. 11) at the current loop address A_L and the old A_I is set as the value of the loop address A_L (S1410). The following processing operations in steps S1410 to S1413 are repeated. More specifically, the old A_I value is sequentially incremented in step S1413, and differential waveform data on the external memory 116 (Fig. 1) designated by the incremented old A_I values read out as data D. The data D are accumulated on the sample data X_P in step S1412. When old A_I value becomes equal to the integral part A_I of the new current address, the sample data X_P has a value corresponding to the integral part A_I of the new current address after loop processing.

When the sample data Xp corresponding to the integral part A_I of the new current address is obtained in this manner, YES is determined in step S1411, and the control enters the above-mentioned arithmetic processing of the interpolation value (S1414).

As described above, waveform data by the DPCM method for one tone generation channel is generated.

Sound Source Processing Based on FM Method (Part 1)

The sound source processing based on the FM method will be described below.

In the FM method, hardware or software elements having the same contents, called "operators", as indicated by OP1 to OP4 in Figs. 18A to 18D, are normally used, and are connected based on connection rules indicated by algorithms 1 to 4 in Figs. 18A to 18D, thereby generating musical tones. In this embodiment, the FM method is realized by a software program.

The operation of this embodiment executed when the sound source processing is performed using two operators will be described below with reference to the operation flow chart shown in Fig. 16A. The algorithm of the processing is shown in Fig. 16B. Variables in the flow chart are FM format data in Table 1 shown in Fig. 11, which data are stored in the corresponding tone generation channel area (Fig. 9) on the RAM 206 or 306 of the MCPU 101 or the SCPU 102.

First, processing of an operator 2 (OP2) as a modulator is performed. In pitch processing (processing for accumulating pitch data for determining an incremental width of an address for reading out waveform data stored in the waveform memory 116), since no waveform data interpolation is performed unlike in the PCM method, an address consists of an integral address A_2, and has no decimal address. Further, modulation waveform data are stored in the external memory 116 (Fig. 1) at sufficiently fine incremental widths.

Pitch data P_2 is added to the present address A_2 (S1601).

A feedback output F_{O2} is added to the address A_2 as a modulation input to obtain a new address A_{M2} which corresponds to phase of a sine wave (S1602). The feedback output F_{O2} has already been obtained upon execution of processing in step S1605 (to be described later) at the immediately preceding interrupt timing.

The value of a sine wave corresponding to the address A_{M2} is calculated. In practice, sine wave data are stored in the external memory 116 (Fig. 1), and are obtained by addressing the external memory 116 by the address A_{M2} to read out the corresponding data (S1603).

Subsequently, the sine wave data is multiplied with an envelope value E_2 to obtain an output O_2 - (S1604).

Thereafter, the output O_2 is multiplied with a feedback level F_{L2} to obtain a feedback output F_{O2} (S1605). This output F_{O2} serves as an input to the operator 2 (OP2) at the next interrupt timing.

The output O_2 is multiplied with a modulation level M_{L2} to obtain a modulation output M_{O2} - (S1606). The modulation output M_{O2} serves as a modulation input to an operator 1 (OP1).

The control then enters processing of the operator 1 (OP1). This processing is substantially the same as that of the operator 2 (OP2) described above, except that there is no modulation input based on the feedback output.

The current address A_1 of the operator 1 is added to pitch data P_1 (S1607), and the sum is added to the above-mentioned modulation output M_{O2} to obtain a new address A_{M1} (S1608).

The value of sine wave data corresponding to this address A_{M1} (phase) is read out from the external memory 116 (Fig. 1) (S1609), and is multiplied with an envelope value E_1 to obtain a musical tone waveform output O_1 (S1610).

The output O_1 is added to a value held in the buffer B (Fig. 12) in the RAM 206 (Fig. 2) or the RAM 306 (Fig. 3) (S1611), thus completing the FM processing for one tone generation channel.

Sound Source Processing Based on TM (Triangular Wave Modulation) Method (Part 1)

The sound source processing based on the TM method will be described below.

The principle of the TM method will be described below.

The FM method is based on the following formula:

$e = A \bullet \sin\{\omega_c t + I(t) \bullet \sin\omega_m t\}$

where $\omega_c t$ is the carrier wave phase angle (carrier signl) $\sin\omega_m t$ is the modulation wave phase angle (modulation signal), and I(t) is the modulation index.

In contrast to this, a phase modulation method called the TM method in this embodiment is based on the following formula:

$e = A \bullet f_T\{f_c(t) + I(t) \bullet \sin\omega_m t\}$

where $f_T(t)$ is the triangular wave function, and is defined by the following functions in units of phase angle regions (where ω is the input):

$f_T(\omega) = 2/\pi \bullet \omega$

.. (region : $0 \leq \omega \leq \pi/2$)

$f_T(\omega) = -1 + 2/\pi(3\pi/2 - \omega)$

.. (region : $\pi/2 \leq \omega \leq 3\pi/2$)

$f_T(\omega) = -1 + 2/\pi(\omega - 3\pi/2)$

.. (region : $3\pi/2 \leq \omega \leq 2\pi$)

f_c is called a modified sine wave, and is the carrier signal generation function obtained by accessing the external memory 116 (Fig. 1) for storing different sine waveform data by the carrier phase angle $\omega_c t$ in units of phase angle regions. f_c of each phase angle region is defined as follows:

$f_c(t) = \pi/2\sin\omega_c t$

.. (region : $0 \leq \omega t \leq \pi/2$)

$f_c(t) = \pi - \pi/2\sin\omega_c t$

.. (region : $\pi \leq \omega t \leq 3\pi/2$)

$f_c(t) = 2\pi + \pi/2\sin\omega_c t$

.. (region : $3\pi/2 \leq \omega_c t \leq 2\pi$). (where n is an integer)

In the TM method, the above-mentioned triangular wave function is modulated by a sum signal obtained by adding a carrier signal generated by the avove-mentioned function $f_c(t)$ to the moulation signal $\sin\omega_m(t)$ at a ratio indicated by the modulation index I(t). In this manner, when the value of the moduation index is "0", a sine wave can be generated, and as the value I(t) is increased, a deeply modulated waveform can always be generated. Various other signals may be used in place of the modulation signal $\sin\omega_m(t)$, and as will be described later, the output of the operator, itself, may be fed back at a predetermined feedback level, or an output from another operator may be input.

The sound source processing based on the TM method according to the above-mentioned principle will be described below with reference to the operation flow chart shown in Fig. 17A. In this case, the sound source processing is also performed using two operators like in the FM method shown in Figs. 16A and 16B, and the algorithm of the processing is shown in Fig. 17B. Variables in the flow chart are TM format data in Table 1 shown in Fig. 11, which data are stored in the corresponding tone generation channel area (Fig. 9) on the RAM 206 or 306 of the MCPU 101 or the SCPU 102.

First, processing of an operator 2 (OP2) as a modulator is performed. In pitch processing, since no waveform data interpolation is performed unlike in the PCM method, an address for addressing the external memory 116 consists of only an integral address A_2.

The current address A_2 is added to pitch data P_2 (S1701).

A modified sine wave corresponding to the address A_2 (phase) is read out from the external memory 116 (Fig. 1) by the modified sine conversion f_c, and is output as a carrier signal O_2 - (S1702).

Subsequently, a feedback output F_{O2} (S1760) as a modulation signal, is added to the carrier signal O_2, and the sum signal is output as a new address O_2 (S1703). The feedback output F_{O2} has already been obtained upon execution of processing in step S1706 (to be described later) at the immediately preceding interrupt timing.

The value of a triangular wave corresponding to the address O_2 is calculated. In practice, triangular wave data are stored in the external memory 116 (Fig. 1), and are obtained by addressing the external memory 116 by the address O_2 to read out the corresponding data (S1704).

Subsequently, the triangular wave data is multiplied with an envelope value E_2 to obtain an output O_2 (S1705).

Thereafter, the output O_2 is multiplied with a feedback level F_{L2} to obtain a feedback output F_{O2} (S1707). In this embodiment, the output F_{O2} serves as an input to the operator 2 (OP2) at the next interrupt timing.

The output O_2 is multiplied with a modulation level M_{L2} to obtain a modulation output M_{O2} - (S1707). The modulation output M_{O2} serves as a modulation input to an operator 1 (OP1).

The control then enters processing of the operator 1 (OP1). This processing is substantially the same as that of the operator 2 (OP2) described above, except that there is no modulation input based on the feedback output.

The current address A_1 of the operator 1 is added to pitch data P_1 (S1708), and the sum is subjected to the above-mentioned modified sine conversion to obtain a carrier signal O_1 (S1709).

The carrier signal O_1 is added to the modulation output M_{O2} to obtain a new value O_1 (S1710), and the value O_1 is subjected to triangular wave conversion (S1711). The converted value is multiplied with an value E_1 to obtain a musical tone waveform output O_1 (S1712).

The output O_1 is added to a value held in the buffer B (Fig. 12) in the RAM 206 (Fig. 2) or the RAM 306 (Fig. 3), thus completing the TM processing for one tone generation channel.

The sound source processing operations based on four methods, i.e., the PCM, DPCM, FM, and TM methods have been described. The FM and TM methods are modulation methods, and, in the above examples, two-operator processing operations are executed based on the algorithms shown in Figs. 16B and 17B. However, in sound source processing in an actual performance, more operators are used, and the algorithms are more complicated. Figs. 18A to 18D show examples. In an algorithm 1 shown in Fig. 18A, four modulation operations including a feedback input are performed, and a complicated waveform can be obtained. In each of algorithms 2 and 3, two sets of algorithms each having a feedback input are arranged parallel to each other, and these algorithms are suitable for expressing a change in tone color during, e.g., transition from an attack portion to a sustain portion. An algorithm 4 has a feature close to a sine wave synthesis method.

The sound source processing operations based on the FM and TM methods using four operators shown in Figs. 18A to 18D will be described below in turn with reference to Figs. 19 and 20.

<u>Sound Source Processing Based on FM Method (Part 2)</u>

Fig. 19 is an operation flow chart of normal sound source processing based on the FM method corresponding to the algorithm 1 shown in Fig. 18A. Variables in the flow chart are stored in the corresponding tone generation channel area (Fig. 9) on the RAM 206 or 306 of the MCPU 101 or the SCPU 102. Although the variables used in Fig. 19 are not the same as data in the FM format of Table 1 in Fig. 11, they are obtained by expanding the concept of the data format shown in Fig. 11, and only have different suffixes.

First, the present address A_4 of an operator 4 (OP4) is added to pitch data P_4 (S1901). The address A_4 is added to a feedback output F_{O4} - (S1905) as a modulation input to obtain a new address A_{M4} (S1902).Furthermore, the value of a sine wave corresponding to the address $_{M4}$ (phase) is read out from the external memory 116 (Fig. 1) (S1903), and is multiplied with an envelope value E_4 to obtain an output O_4 (S1904). Thereafter, the output O_4 is multiplied with a feedback level F_{L4} to obtain a feedback output F_{O4} (S1905). The output O_4 is multiplied with a modulation level M_{L4} to obtain a modulation output M_{O4} (S1906). The modulation output M_{O4} serves as a modulation input to the next operator 3 (OP3).

The control then enters processing of the operator 3 (OP3). This processing is substantially the same as that of the operator 4 (OP4) described above, except that there is no modulation input based on the feedback output. The current address A_3 of the operator 3 (OP3) is added to pitch data P_3 to obtain a new current address A_3 (S1907). The address A_3 is added to a modulation output M_{O4} as a modulation input, thus obtaining a new address A_{M3} (S1908). Furthermore, the value of a sine wave corresponding to the address A_{M3} - (phase) is read out from the external memory 116 (Fig. 1) (S1909), and is multiplied with an envelope value E_3 to obtain an output O_3 (S1910). Thereafter, the output O_3 is multiplied with a modulation level M_{L3} to obtain a modulation output M_{O3} - (S1911). The modulation output M_{O3} serves as a modulation input to the next operator 2 (OP2).

Processing of the operator 2 (OP2) is then executed. However, this processing is substantially the same as that of the operator 3, except that a modulation input is different, and a detailed description thereof will be omitted.

Finally, the control enters processing of an operator 1 (OP1). In this case, the same processing operations as described above are performed up to step S1920. A musical tone waveform output O_1

obtained in step S1920 is added to data stored in the buffer B as a carrier (S1921).

Sound Source Processing Based on TM Method (Part 2)

Fig. 20 is an operation flow chart of normal sound source processing based on the TM method corresponding to the algorithm 1 shown in Fig. 18A. Variables in the flow chart are stored in the corresponding tone generation channel area (Fig. 9) on the RAM 206 or 306 of the MCPU 101 or the SCPU 102. Although the variables used in Fig. 19 are not the same as data in the TM format of Table 1 in Fig. 11, they are obtained by expanding the concept of the data format shown in Fig. 11, and only have different suffixes.

The present address A_4 of the operator 4 (OP4) is added to pitch data P_4 (S2061). A modified sine wave to the above-mentioned address A_4 (phase) is read out from the external memory 116 (Fig. 1) by the modified sine conversion f_c, and is output as a carrier signal O_4 (S2002). A feedback output F_{O4} (see S2007) as a modulation signal is added to the carrier signal O_4, and the sum signal is output as a new address O_4 (S2003). The value of a triangular wave corresponding to the address O_4 (phase) is read out from the external memory 116 (Fig. 1) (to be referred to as a triangular wave conversion hereinafter) (S2004), and is multiplied with an envelope value E_4, thus obtaining an output O_4 (S2005). Thereafter, the output O_4 is multiplied with a modulation level M_{L4} to obtain a modulation output M_{O4} (S2006). The output O_4 is multiplied with a feedback level F_{L4} to obtain a feedback output F_{O4} (S2007). The modulation output M_{O4} serves as a modulation input to the next operator 3 (OP3).

The control then enters processing of the operator 3 (OP3). This processing is substantially the same as that of the operator 4 (OP4) described above, except that there is no modulation input based on the feedback output. The current address A_3 of the operator 3 (OP3) is added to pitch data P_3 (S2008) and the sum is subject to modified sine conversion to obtain a carrier signal O_3 (S2009). The carrier signal O_3 is added to the above-mentioned modulation output M_{O4} to obtain a new value O_3 (S2010), and the value O_3 is subject to triangular wave conversion (S2011). The converted value is multiplied with an envelope value E_3 to obtain an output O_3 (S2012). The output O_3 is multiplied with a modulation level M_{L3} to obtain a modulation output M_{O3} (S2013). The modulation output M_{O3} serves as a modulation input to the next operator 2 (OP2).

Processing of the operator 2 (OP2) is then executed. However, this processing is substantially the same as that of the operator 3, except that a modulation input is different, and a detailed description thereof will be omitted.

Finally, the control enters processing of an operator 1 (OP1). In this case, the same processing operations as described above are performed up to step S2024. A musical tone waveform output O_1 obtained in step S2024 is accumulated in the buffer B (Fig. 12) as a carrier (S2025).

The embodiment of the normal sound processing operations based on the modulation methods has been described. However, the above-mentioned processing is for one tone generation channel, and in practice, the MCPU 101 and the SCPU 102 each execute processing for eight channels (Fig. 4D). If a modulation method is designated in a given tone generation channel, the above-mentioned sound source processing based on the modulation method is executed.

Modification of Modulation Method (Part 1)

The first modulation of the sound source processing based on the modulation method will be described below.

The basic concept of this processing is shown in the flow chart of Fig. 21.

In Fig. 21, operator 1, 2, 3, and 4 processing operations have the same program architecture although they have different variable names to be used.

Each operator processing cannot be executed unless a modulation input is determined. This is because a modulation input to each operator processing varies depending on the algorithm, as shown in Figs. 18A to 18D. Which operator processing output is used as a modulation input or whether or not an output from its own operator processing is fed back, and is used as its own modulation input in place of another operator processing must be determined. In the operation flow chart shown in Fig. 21, such determinations are simultaneously performed in algorithm processing (S2105), and the connection relationship obtained by this processing determine modulation inputs to the respective operator processing operations (S2102 to S2104). Note that a given initial value is set as an input to each operator processing at the beginning of tone generation.

When the operator processing and the algorithm processing are separated in this manner, the program of the operator processing can remain the same, and only the algorithm processing can be modified in correspondence with algorithms. Therefore, the program size of the overall sound source processing based on the modulation method can be greatly reduced.

A modification of the FM method based on the above-mentioned basic concept will be described below. The operator 1 processing in the operation flow chart showing operator processing based on the FM method in Fig. 21 is shown in Fig. 22A, and an arithmetic algorithm per operator is shown in Fig. 22B. The remaining operator 2 to 4 processing operations are the same except for different suffix numbers of variables. Variables in the flow chart are stored in the corresponding tone generation channel (Fig. 9) on the RAM 206 or 306 of the MCPU 101 or the SCPU 102.

An address A_1 corresponding to a phase angle is added to pitch data P_1 to obtain a new address A_1 (S2201). The address A_1 is added to a modulation input M_{I1}, thus obtaining an address A_{M1} - (S2202). The modulation input M_{I1} is determined by the algorithm processing in step S2105 (Fig. 21) at the immediately preceding interrupt timing, and may be a feed back output F_{O1} of its own operator, or an output M_{O2} from another operator, e.g., an operator 2 depending on the algorithm. The value of a sine wave corresponding to this address (phase) A_{M1} is read out from the external memory 116 (Fig. 1), thus obtaining an output O_1 (S2203). Thereafter, a value obtained by multiplying the output O_1 with envelope data E_1 serves as an output O_1 of the operator 1 (52204). The output O_1 is multiplied with a feedback level F_{L1} to obtain a feedback output F_{O1} (S2205). The output O_1 is multipled with a modulation level M_{L1}, thus obtaining a modulation output M_{O1} (S2206).

A modification of the TM method based on the above-mentioned basic concept will be described below. The operator 1 processing in the operation flow chart showing operator processing based on the TM method in Fig. 21 is shown in Fig. 23A, and an arithmetic algorithm per operator is shown in Fig. 23B. The remaining operator 2 to 4 processing operations are the same except for different suffix numbers of variables. Variables in the flow chart are stored in the corresponding tone generation channel (Fig. 9) on the RAM 206 or 306 of the MCPU 101 or the SCPU 102.

The current address A_1 is added to pitch data P_1 (S2301). A modified sine wave corresponding to the above-mentioned address A_1 (phase) is read out from the external memory 116 (Fig. 1) by the modified sine conversion f_c, and is generated as a carrier signal O_1 (S2302). The output O_1 is added to a modulation input M_{I1} as a modulation signal, and the sum is defined as a new address O_1 - (S2303). The value of a triangular wave corresponding to the address O_1 (phase) is read out from the external memory 116 (S2304), and is multiplied with an envelope value E_1 to obtain an output O_1 (S2306). Thereafter, the output O_1 is multiplied with a feedback level F_{L1} to obtain a feedback output F_{O1} (S2306). The output O_1 is multiplied with a modulation level M_{L1} to obtain a modulation output M_{O1} (S2307).

The algorithm processing in step S2105 in Fig. 21 for determining a modulation input in the operator processing in both the above-mentioned modulation methods, i.e., the FM and TM methods will be described in detail below with reference to the operation flow chart of Fig. 24. The flow chart shown in Fig. 24 is common to both the FM and TM methods, and the algorithms 1 to 4 shown in Figs. 18A to 18D are selectively processed. In this case, choices of the algorithms 1 to 4 are made based on an instruction (not shown) from a player (S2400).

The algorithm 1 is of a series four-operator (to be abbreviated to as an OP hereinafter) type, and only the OP4 has a feedback input. More specifically, in the algorithm 1,

a feedback output F_{O4} of the OP4 serves as the modulation input M_{I4} of the OP4 (S2401),

a modulation output M_{O4} of the OP4 serves as a modulation input M_{I3} of the OP3 (S2402),

a modulation output O_{P3} of the OP3 serves as a modulation input M_{I2} of the OP2 (S2403),

a modulation output M_{O2} of the OP2 serves as a modulation input M_{I1} of the OP1 (S2404), and

an output O_1 from the OP1 is added to the value held in the buffer B (Fig. 12) as a carrier output (S2405).

In the algorithm 2, as shown in Fig. 18B, the OP2 and the OP4 have feedback inputs. More specifically, in the algorithm 2,

a feedback output F_{O4} of the OP4 serves as a modulation input M_{I4} of the OP4 (S2406),

a modulation output M_{O4} of the OP4 serves as a modulation input M_{I3} of the OP3 (S2407),

a feedback output F_{O2} of the OP2 serves as a modulation input M_{I2} of the OP2 (S2408),

modulation outputs M_{O2} and M_{O3} of the OP2 and serve as a modulation input M_{I1} of the OP1 (S2409), and

an output O_1 from the OP1 is added to the value held in the buffer B as a carrier output (S2410).

In the algorithm 3, the OP2 and OP4 have feedback inputs, and two modules in which two operators are connected in series with each other are connected in parallel with each other. More specifically, in the algorithm 3,

a feedback output F_{O4} of the OP4 serves as a modulation input M_{I4} of the OP4 (S2411),

a modulation output M_{O4} of the OP4 serves as a modulation input M_{I3} of the OP3 (S2412),

a feedback output F_{O2} of the OP2 serves as a modulation input M_{I2} of the OP2 (S2413),

a modulation output M_{O2} of the OP2 serves as a modulation input M_{I1} of the OP1 (S2414), and

outputs O_1 and O_3 from the OP1 and OP3 are added to the value held in the buffer B as carrier outputs (S2415).

The algorithm 4 is of a parallel four-OP type, and all the OPs have feedback inputs. More specifically, in the algorithm 4,

a feedback output F_{O4} of the OP4 serves as a modulation input M_{I4} of the OP4 (S2416),

a feedback output F_{O3} of the OP3 serves as a modulation input M_{I3} of the OP3 (S2417),

a feedback output F_{O2} of the OP2 serves as a modulation input M_{I2} of the OP2 (S2418),

a feedback output F_{O1} of the OP1 serves as an input M_{I1} of the OP1 (S2419), and

outputs O_1, O_2, O_3, and O_4 from all the OPs are added to the value held in the buffer B (S2420).

The sound source processing for one channel is completed by the above-mentioned operator processing and algorithm processing, and tone generation (sound source processing) continues in this state unless the algorithm is changed.

Modification of Modulation Method (Part 2)

The second modification of the sound source processing based on the modulation method will be described below.

In the various modulation methods described above, processing time is increased as the complicated algorithms are programmed, and as the number of tone generation channels (the number of polyphonic channels) is increased.

In the second modification to be described below, the first modification shown in Fig. 21 is further developed, so that only operator processing is performed at a given interrupt timing, and only algorithm processing is performed at the next interrupt timing. Thus, the operator processing and the algorithm processing are alternately executed. In this manner, a processing load per interrupt timing can be greatly reduced. As a result, one sample data per two interrupts is output.

This operation will be described below with reference to the operation flow chart shown in Fig. 25.

In order to alternately execute the operator processing and the algorithm processing, whether or not a variable S is zero is checked (S2501). The variable is provided for each tone generation channel, and is stored in the corresponding tone generation channel area (Fig. 9) on the RAM 206 or 306 of the MCPU 101 or the SCPU 102.

If S = 0 at a given interrupt timing, the process enters an operator processing route, and sets the variable S to a value "1" (S2502). Subsequently, operator 1 to 4 processing operations are executed (S2503 to S2506). This processing is the Same as that in Fig. 22 or 23.

The process exits from the operator processing route, and executes output processing for setting a value of the buffer BF (for the FM method) or the buffer BT (for the TM method) (S2510). The buffer BF or BT is provided for each tone generation channel, and is stored in the corresponding tone generation channel area (Fig. 9) on the RAM 206 or 306 of the MCPU 101 or the SCPU 102. The buffer BF or BT stores a waveform output value after the algorithm processing. At the current interrupt timing, however, no algorithm processing been executed, and the content of the buffer BF or BT is not updated. For this reason, the same waveform output value as that at the immediately preceding interrupt timing is output.

With the above processing, sound source processing for one tone generation channel at the current interrupt timing is completed. In this case, data obtained by the current operator 1 to 4 processing operations are stored in the corresponding tone generation channel area (Fig. 9) on the RAM 206 or 306 of the MCPU 101 or the SCPU 102.

At the next interrupt timing, since the variable S is set to be 1 at the immediately preceding interrupt timing, the flow advances to step S2507. The process then enters an algorithm processing route, and sets the variable S to be a value "0". Subsequently, the algorithm processing is executed (S2508).

In this processing, the data processed in the operator 1 to 4 processing operations at the immediately preceding interrupt timing and stored in the corresponding tone generation channel area (Fig. 9) are used, and processing for determining a modulation input for the next operator processing is executed. In this processing, the content of the buffer BF or BT is rewritten, and a waveform output value at that interrupt timing can be obtained. The algorithm processing is shown in detail in the operation flow chart of Fig. 26. In this flow chart, the same processing operations as in Fig. 24 are executed in steps denoted by the same reference numerals as in Fig. 24. A difference between Figs. 24 and 26 is an output portion in steps S2601 to S2604. In the case of algorithms 1 and 2, the content of the output O_1 of the operator 1 processing is directly stored in the buffer BF or BT (S2601 and S2602). In the case of the algorithm 3, a value as a sum of the outputs O_1 and O_3 is stored in the buffer BF or BT (S2603). Furthermore, in the case of the algorithm 4, a value as a sum of the output O_1 and the outputs O_2, O_3, and O_4 is stored in the buffer BF or BT (S2604).

As described above, since the operator processing and the algorithm processing are alternately executed at every other interrupt timing, a processing load per interrupt timing of the sound source processing program can be remarkably decreased. In this case, since an interrupt period need not be prolonged, the processing load can be reduced without increasing an interrupt time of the main operation flow chart shown in Fig. 4A, i.e., without influencing the program operation. Therefore, a keyboard key sampling interval executed in Fig. 4A will not be prolonged, and the response performance of an electronic musical instrument will not be impaired.

The operations for generating musical tone data in units of tone generation channels by the software sound source processing operations based on various sound source methods have been described.

Function Key Processing

The operation of the function key processing (S403) in the main operation flow chart shown in Fig. 4A when an actual electronic musical instrument is played will be described in detail below.

In the above-mentioned sound source processing executed for each tone generation channel, parameters corresponding to sound source methods are set in the formats shown in Fig. 11 in the corresponding tone generation channel area (Fig. 9) on the RAM 206 or 306 (Figs. 2 and 3) by one of the function keys 801 (Fig. 8A) connected to the operation panel of the electronic musical instrument via the input port 210 (Fig. 2) of the MCPU 101.

Fig. 27 shows an arrangement of some function keys 801 shown in Fig. 8A. In Fig. 27, some function keys 801 are realized as tone color switches. When one of switches "piano", "guitar",..., "koto" in a group A is depressed, a tone color of the corresponding instrument tone is selected, and a guide lamp is turned on. Whether the tone color of the selected instrument tone is generated in the DPCM method or the TM method is selected by a DPCM/TM switch 2701.

On the other hand, when a switch "tuba" in a group B is depressed, a tone color based on the FM method is designated; when a switch "bass" is depressed, a tone color on both the PCM and TM methods is designated; and when a switch "trumpet" is depressed, a tone color based on the PCM method is designated. Then, a musical tone based on the designated sound source method is generated.

Figs. 28A and 28B show of sound source methods to the respective tone generation channel region (Fig. 9) on the RAM 206 or 306 when the switches "piano" and "bass" are depressed. When the switch "piano" is depressed, the DPCM method is assigned to all the 8-tone polyphonic tone generation channels of the MCPU 101 and the SCPU 102, as shown in Fig. 28A. When the switch "bass" is depressed, the PCM method is assigned to the odd-numbered tone generation channels, and the TM method is assigned to the even-numbered tone generation channels, as shown in Fig. 28B. Thus, a musical tone waveform for one musical tone can be obtained by mixing tone waveforms generated in the two tone generation channels based on the PCM and TM methods. In this case, a 4-tone polyphonic system per CPU is attained, and an 8-tone polyphonic system as a total of two CPUs is attained.

Fig. 29 is a partial operation flow chart of the function key processing in step S403 in the main operation flow chart shown in Fig. 4A, and shows processing corresponding to the tone color designation switch group shown in Fig. 27.

It is checked if a player operates the DPCM/TM switch 2701 (S2901). If YES in step S2901, it is checked if a variable M is zero (S2902). The variable M stored on the RAM 206 (Fig. 2) of the MCPU 101, and has a value "0" for the DPCM method; a value "1" for the TM method. If YES in step S2902, i.e., if it is determined that the value of the variable M is 0, the variable M is set to be a value "1" (S2903). This means that the DPCM/TM switch 2701 is depressed in the DPCM method selection state, and the selection state is changed to the TM method selection state. However, if NO in step S2902, i.e., if it is determined that the value of the variable M is "1", the variable M is set to be a value "0" (S2904). This means that the DPCM/TM switch 2701 is depressed in the TM method selection state, and the selection state is changed to the DPCM method selection state.

It is checked if a tone color in the group A shown in Fig. 27 is currently designated (S2905). Since the DPCM/TM switch 2701 is valid for tone colors of only group A, only when a tone color in the group A is designated, and YES is determined in step S2905, operations corresponding to the DPCM/TM switch 2701 in steps S2906 to S2908 are executed.

It is checked if the variable M is "0" (S2906).

If YES in step S2906, since the DPCM method is selected by the DPCM/TM switch 2701, DPCM data are set in the DPCM format shown in Fig. 11 in the corresponding tone generation channel areas on the RAMs 206 and 306 (Figs. 2 and 3). More specifically, sound source method No. data G indicating the DPCM method is set in the start area of the corresponding tone generation channel area (see the column of DPCM in Fig. 11). Subsequently, various parameters corresponding to currently designated tone colors are respectively set in the second and subsequent areas of the corresponding tone generation channel area (S2907).

If NO in step S2906, since the TM method is selected by the DPCM/TM switch 2701, TM data are set in the TM format shown in Fig. 11 in the corresponding generation channel areas. More specifically, sound source method No. data G indicating the TM method is set in the start area of the corresponding tone generation channel area. Subsequently, various parameters corresponding to currently designated tone colors are respectively set in the second and subsequent areas of the corresponding tone generation channel area (S2908).

A case has been exemplified wherein the DPCM/TM switch 2701 shown in Fig. 27 is operated. If the switch 2701 is not operated and NO is determined in step S2901, or if tone color of the group A is not designated and NO is determined in step S2905, processing from step S2909 is executed.

It is checked in step S2909 if a change in tone color switch shown in Fig. 27 is detected.

If NO in step S2909, since processing for the tone color switches need not be executed, the function key processing (S403 in Fig. 4A) is ended.

If it is determined that a change in tone color switch is detected, and YES is determined in step S2909, it is checked if a tone color in the group B is designated (S2910).

If a tone color in the group B is designated, and YES is determined in step S2910, data for the sound source method corresponding to the designated tone color are set in the predetermined format in the corresponding tone generation channel areas on the RAMs 206 and 306 (Figs. 2 and 3). More specifically, sound source method No. data G indicating the sound source method is set in the start area of the corresponding tone generation channel area (Fig. 11). Subsequently, various parameters corresponding to the currently designated tone color are respectively set in the second and subsequent areas of the corresponding tone generation channel area (S2911). For example, when the switch "bass" in Fig. 27 is selected, data corresponding to the PCM method are set in the odd-numbered tone generation channel areas, and data corresponding to the TM method are set in the even-numbered tone generation channel areas.

If it is determined that the tone color switch in the group A is designated, and NO is determined in step S2910, it is checked if the variable M is "1" (S2912). If the TM method is currently selected, and YES is determined in step S2912, data are set in the TM format (Fig. 11) in the corresponding tone generation channel area (S2913) like in step S2908 described above.

If the DPCM method is selected, and NO is determined in step S2912, data are set in the DPCM format (Fig. 11) in the corresponding tone generation channel area (S2914) like in step S2907 described above.

First Embodiment of ON Event Keyboard Key Processing

The operation of the keyboard key processing (S405) in the main operation flow chart shown in Fig. 4A executed when an actual electronic musical instrument is played will be described below.

The first embodiment of ON event keyboard key processing will be described below.

In this embodiment, when a tone color in the group A shown in Fig. 27 is designated, the sound source method to be set in the corresponding tone generation channel area of the RAM 206 or 306 (Figs. 2 and 3) is automatically switched in accordance with an ON key position, i.e., a tone range of a musical tone. This embodiment has a boundary between key code numbers 31 and 32 on the keyboard shown in Fig. 8B. That is, when a key code of an ON key falls within a bass tone range equal to or lower than the 31st key code, the DPCM method is assigned to the corresponding tone generation channel. On the other hand, when a key code of an ON key falls within a high tone range equal to or higher than the 32nd key code, the TM method is assigned to the corresponding tone generation channel. When a tone color in the group B in Fig. 27 is designated, no special keyboard key processing is executed.

Fig. 30 is a partial operation flow chart of the keyboard key processing in step S405 in the main operation flow chart of Fig. 4A.

It is checked if a tone color in the group A is presently designated (S3001).

If NO in step S3001, and a tone color in the group B is currently designated, special processing in Fig. 30 is not performed.

If YES in step S3001, and a tone color in the group A is currently designated, it is checked if a key code of a key which is detected as an "ON key" in the keyboard key scanning processing in step S404 in the main operation flow chart shown in Fig. 4A is equal to or lower than the 31st key code (S3002).

If a key in the bass tone range equal to or lower than the 31st key code is depressed, and YES is determined in step S3002, it is checked if the variable M is "1" (S3003). The variable M is set in the operation flow chart shown in Fig. 29 as a part of the function key processing in step S403 in the main operation flow chart shown in Fig. 4A, and is "0" for the DPCM method; "1" for the TM method, as described above.

If YES (M = "1") in step S3003, i.e., if it is determined that the TM method is currently designated as the sound source method, DPCM data in Fig. 11 are set in a tone generation channel area of the RAM 206 or 306 (Figs. 2 and 3) where the ON key is assigned so as to change the TM method to the DPCM method as a sound source method for the bass tone range (see the column of DPCM in Fig. 11). More specifically, sound source method No. data G indicating the DPCM method is set in the start area of the corresponding tone generation channel area. Subsequently, various parameters corresponding to the currently designated tone color are respectively set in the second and subsequent areas of the corresponding tone generation channel area (S3004). Thereafter, a value "1" is set in a flag C (S3005). The flag C is a variable (Fig. 11) stored in each tone generation channel area on the RAM 206 (Fig. 2) of the MCPU 101, and is used in OFF event processing to be described later with reference to Fig. 32.

If it is determined that a key in the high tone range equal to or higher than the 31st key code is depressed, and NO is determined in step S3002, it is checked if the variable M is "1" (S3006).

If NO (M = "0") in step S3006, i.e., if it is determined that the DPCM method is currently designated as the sound source method, TM data in Fig. 11 are set in a tone generation channel area of the RAM 206 or 306 (Figs. 2 and 3) where the ON key is assigned so as to change the DPCM method to the TM method as a sound source method for the high tone range (see the column of TM in Fig. 11). More specifically, sound source method No. data G indicating the TM method is set in the start area of the corresponding tone generation channel area. Subsequently, various parameters corresponding to the currently designated tone color are respectively set in the second and subsequent areas of the corresponding tone generation channel area (S3007). Thereafter, a value "2" is set in a flag C (S3008).

In the above-mentioned processing, if NO in step S3003 and if YES in step S3006; since the desired sound source method is originally selected, no special is executed.

Second Embodiment of ON Event Keyboard Key Processing

The second embodiment of the ON event keyboard key processing will be described below.

In the second embodiment of the ON event keyboard key processing, when a tone color in the group A in Fig. 27 is designated, a sound source method to be set in the corresponding tone generation channel area (Fig. 9) on the RAM 206 or 206 (Figs. 2 and 3) of the MCPU 101 or the SCPU 102 is automatically switched in accordance with an ON key speed, i.e., a velocity. In this case, a switching boundary is set at a velocity value "64" half the maximum value "127" of the MIDI (Musical Instrument Digital Interface) standards. That is, when the velocity value of an ON key is equal to or larger than 64, the DPCM method is assigned; when the velocity of an ON key is equal to or smaller than 64, the TM method is assigned. When a tone color in the group B in Fig. 27 is designated, no special keyboard key processing is executed.

Fig. 31 is a partial operation flow chart of the keyboard key processing in step S405 in the main operation flow chart shown in Fig. 4A.

It is checked if a tone color in the group A in Fig. 27 is currently designated (S3101).

If NO in step S3101, and a tone color in the group B is presently selected, the special processing in Fig. 30 is not executed.

If YES in step S3101, and a tone color in the group A is presently selected, it is checked if the velocity of a key which is detected as an "ON key" in the keyboard key scanning processing in step S404 in the main operation flow Chart Shown in Fig. 4A is equal to or larger than 64 (S3102). Note that the velocity value "64" corresponds to "mp (mezzo piano)" of the MIDI standards.

If it is determined that the velocity value is equal to or larger than 64, and YES is determined in step S3102, it is checked if the variable M is "1" (S3102). The variable M is set in the operation flow chart shown in Fig. 29 as a part of the function key processing in step S403 in the main operation flow chart shown in Fig. 4A, and is "0" for the DPCM method; "1" for the TM method, as described above.

If YES (M = "1") in step S3103, and the TM method is currently designated as the sound source method, DPCM data in Fig. 11 are set in a tone generation channel area of the RAM 206 or 306 (Figs. 2 and 3) where the ON key is assigned so as to change the TM method to the DPCM method as a sound source method for a fast ON key operation (S3104), and a value "1" is set in the flag C (S3105).

If it is determined that the velocity value is smaller than 64 and NO is determined in step S3102, it is further checked if the variable M is "1" (S3106).

NO (M = "0") in step S3106, and the DPCM method is currently designated as the sound source method, TM data in Fig. 11 are set in a tone generation channel area of the RAM 206 or 306 where the ON key is assigned so as to change the DPCM method to the TM method as a sound source method for a slow ON key operation (S3107). Thereafter, a value "2" is set in the flag C (S3108).

In the above-mentioned processing, if NO in step S3103 and if YES in step S3106, since the desired sound source method is originally selected, no special processing is executed.

Embodiment of OFF Event Keyboard Key Processing

The embodiment of the OFF event keyboard key processing will be described below.

According to the above-mentioned ON event keyboard key processing, the sound source method is automatically set in accordance with a key range (tone range) or a velocity. Upon an OFF event, the set sound source method must be restored. The embodiment of the OFF event keyboard key processing to be described below can realize this processing.

Fig. 32 is a partial operation flow chart of the keyboard key processing in step S405 in the main operation flow chart shown in Fig. 4A.

The value of the flag C set in the tone generation channel area on the RAM 206 or 306 (Figs. 2 and 3), where the key determined as an "OFF key" in the keyboard key scanning processing in step S404 in the main operation flow chart of Fig. 4A is assigned, is checked. The flag C is set in steps S3005 and S3008 in Fig. 30, or in step S3105 or S3108 in Fig. 31, has an initial value "0", is set to be "1" when the sound source method is changed from the TM method to the DPCM method upon an ON event, and is set to be "2" when the sound source method is changed from the DPCM method to the TM method. When the sound source method is left unchanged upon an ON event, the flag C is left at the initial value "0".

If it is determined in step S3201 in the OFF event processing in Fig. 32 that the value of the flag C is "0", since the sound source method is left unchanged in accordance with a key range or a velocity, no special processing is executed, and normal OFF event processing is performed.

If it is determined in step S3201 that the value of the flag C is "1", the sound source method is changed from the TM method to the DPCM method upon an ON event. Thus, TM data in Fig. 11 is set in the tone generation channel area on the RAM 206 or 306 (Fig. 2 or 3) where the ON key is assigned to restore the sound source method to the TM method. More specifically, sound source No. data G indicating the TM method is set in the start area of the corresponding tone generation channel area. Subsequently, various parameters corresponding to the presently designated tone color are respectively set in the second and subsequent areas of the corresponding tone generation channel area (S3202).

If it is determined in step S3201 that the value of the flag C is "2", the sound source method is changed from the DPCM method to the TM method. Thus, DPCM data in Fig. 11 is set in the tone generation channel area on the RAM 206 or 306 where the ON key is assigned to restore the sound source method from the TM method to the DPCM method. More specifically, sound source method No. data G indicating the DPCM method is set in the start area of the corresponding tone generation channel area. Subsequently, various parameters corresponding to the presently designated tone color are respectively set in the second and subsequent areas of the corresponding tone generation channel area (S3203).

After the above-mentioned operation, the value of the flag C is reset to "0", and the processing in Fig. 32 is completed. Subsequently, normal OFF event processing (not shown) is executed.

Other Embodiments

In the above embodiments of the present invention described above, as shown in Fig. 1, the two CPUs, i.e., the MCPU 101 and the SCPU 102 share processing of different tone generation channels. However, the number of CPUs may be one or three or more.

If the control ROMs 201 and 301 shown in Figs. 2 and 3, and the external memory 116 are constituted by, e.g., ROM cards, various sound source methods can be presented to a user by means of the ROM cards.

Furthermore, the input port 210 of the MCPU 101 shown in Fig. 2 can be connected to various other operation units in addition to the instrument operation unit shown in Fig. 8A. Thus, various other electronic musical instruments can be realized. In addition, the present invention may be realized as a sound source module for executing only the sound source processing while receiving performance data from another electronic musical instrument.

Various methods of assigning sound source methods to tone generation channels by the function keys 801 or the keyboard keys 802 in Fig. 8A including those based on tone colors, tone ranges, and velocities, may be proposed.

In addition to the FM and TM methods, the present invention may be applied to various other modulation methods.

In the modulation method, the above embodiment exemplifies a 4-operator system. However, the number of operators is not limited to this.

In this manner, according to the present invention, a musical tone waveform generation apparatus can be constituted by versatile processors without requiring a special-purpose sound source circuit at all. For this reason, the circuit scale of the overall musical tone waveform generation apparatus can be reduced, and the apparatus can be manufactured in the same manufacturing technique as a conventional microprocessor when the apparatus is constituted by an LSI, thus improving the yield of chips. Therefore, manufacturing cost can be greatly reduced. Note that a musical tone signal output unit can be constituted by a simple latch circuit, resulting in almost no increase in manufacturing cost after the output unit is added.

When the modulation method is required to be changed between a phase modulation method and a frequency modulation method, or when the number of polyphonic channels is required to be changed, a sound source processing program to be stored in a program storage means need only be changed to meet the above requirements. Therefore, development cost of a new musical tone waveform generation apparatus can be greatly decreased, and a new sound source method can be presented to a user by means of, e.g., a ROM card.

In this case, since a data architecture for attaining a data link between a performance data processing program and a sound source processing program via musical tone generation data on a data storage means, and a program architecture for executing the sound source processing program at predetermined time intervals while interrupting the performance data processing program are realized, two processors need not be synchronized, and the programs can be greatly simplified. Thus, complicated sound source processing such as the modulation method can be executed with a sufficient margin.

Furthermore, since a change in processing time depending on the type of modulation method or a selected musical tone generation algorithm in the modulation method can be absorbed by a musical tone signal output means, no complicated timing control program for outputting a musical tone signal to, e.g., a D/A converter can be omitted.

Furthermore, the present invention has, as an architecture of the sound source processing program, a processing architecture for simultaneously executing algorithm processing operations as I/O processing among operator processing operations before or after simultaneous execution of at least one operator processing as a modulation processing unit. For this reason, when one of a plurality of algorithms is selected to execute sound source processing, a plurality of types of algorithm processing portions are prepared, and need only be switched as needed. Therefore, the sound source processing program can be rendered very compact. The small program size can greatly contribute to a compact, low-cost musical tone waveform generation apparatus.

Claims

1. A musical tone waveform generation apparatus characterized by comprising:

   storage means (201) for storing a sound source processing program based on a predetermined modulation method;

   musical tone signal generation means (101) for generating a musical tone signal on the basis of a process of the modulation method by executing the sound source processing program stored in said storage means; and

   musical tone signal output means (107, 108) for outputting the musical tone signal generated by said musical tone signal generation means at predetermined time intervals.

2. An apparatus according to claim 1, characterized in that said musical tone signal output means (107, 108) comprises:

   timing signal generating means (203) for generating a timing signal for each predetermined sampling period;

   first latch means (601) for latching a digital musical tone signal generated by said musical tone signal generation means (101) at an outputting timing of the digital musical tone signal from said musical tone signal generation means (101); and

   second latch means (602) for outputting the digital musical tone signal by latching an output signal of said first latch means (601) when the timing signal is generated from said timing signal generating means (203).

3. An apparatus according to claim 1, characterized in that the method is a method of receiving a mixed signal obtained by mixing a carrier signal and a modulation signal as an input and outputting a modulated musical tone signal as an output.

4. An apparatus according to claim 3, characterized in that a functional relationship between the input and the output is expressed by neither of sine and cosine function relationships, and the carrier signal is a signal for making the output to be a sine or cosine wave at a single frequency when the carrier signal is directly used as the input.

5. An apparatus according to claim 3, characterized in that a functional relationship between the input and the output is expressed by a sine function, and the carrier signal is defined by a sine wave.

6. A musical tone waveform generation apparatus characterized by comprising:

   program storage means (201) for storing a performance data processing program for processing performance data, and a sound source processing program, based on a modulation method, for obtaining a musical tone signal;

address control means (205) for controlling an address of said program storage means (201);

data storage means (206) for storing musical tone generation data necessary for generating a musical tone signal based on the modulation method;

arithmetic processing means (208, 209) for performing arithmetic processing;

program execution means (101) for executing the performance data processing program and the sound source processing program stored in said program storage means (201) while controlling said address control means (205), said data storage means (206), and said arithmetic processing means (208, 209), said program execution means normally executing the performance data processing program to control musical tone generation data on said data storage means (206), executing the sound source processing program at predetermined time intervals, executing the performance data processing program again upon completion of the sound source processing program, and generating a musical tone signal by the modulation method on the basis of the musical tone generation data on said data storage means upon execution of said sound source processing program; and

musical tone signal output means (107, 108) for holding the musical tone signal obtained when said program execution means (101) executes the sound source processing program, and outputting the held musical tone signal at predetermined output time intervals.

7. An apparatus according to claim 6, characterized in that said program execution means (101) includes interrupt control means (203) for generating an interrupt signal at the predetermined time intervals, and

said program execution means (101) interrupts the performance data processing program at a generation timing of the interrupt signal from said interrupt control means (203) during execution of the performance data processing program, executes the sound source processing program, cancels the interruption upon completion of the sound source processing program, and restarts execution of the performance data processing program.

8. A musical tone waveform generation apparatus characterized by comprising:

storage means (201) for storing a sound source processing program associated with a modulation method, having an operator processing program for executing operator processings, and an algorithm processing program for executing algorithm processing for determining an input/output relationship among operator processings;

musical tone signal generation means (101) for generating a musical tone signal by executing the operator processing operations based on the operator processing program at a time, and executing the algorithm processing at a time based on the algorithm processing program independently of the operator processing program; and

musical tone signal output means (107, 108) for outputting the musical tone signal generated by said musical tone signal generation means at predetermined output time intervals.

9. An apparatus according to claim 8, characterized in that said musical tone signal output means (107, 108) comprises:

timing signal generating means (203) for generating a timing signal for each predetermined sampling period;

first latch means (601) for latching a digital musical tone signal generated by said musical tone signal generation means (101) at an outputting timing of the digital musical tone signal from said musical tone signal generation means (101); and

second latch means (602) for outputting the digital musical tone signal by latching an output signal of said first latch means (601) when the timing signal is generated from said timing signal generating means (203).

10. An apparatus according to claim 8, characterized in that the modulation method is a method of receiving a mixed signal obtained by mixing a carrier signal and a modulation signal as an input and outputting a modulated musical tone signal as an output.

11. An apparatus according to claim 10, characterized in that a functional relationship between the input and the output is expressed by neither of sine and cosine function relationships, and the carrier signal is a signal for making the output to be a sine or cosine wave at a single frequency when the carrier signal is directly used as the input.

12. An apparatus according to claim 10, characterized in that a functional relationship between the input and the output is expressed by a sine function, and the carrier signal.

13. A musical tone waveform generation apparatus characterized by comprising:

program storage means (201) for storing a performance data processing program for processing performance data, and a sound source processing program based on a modulation method for obtaining a musical tone signal, the sound source processing program having a processing architecture in which algorithm processing operations for determining an input/output relationship among a plurality of operator processing operations are executed at a time after or before execution of the plurality of operator processing operations at a time as modulation processing units;

address control means (205) for controlling an address of said program storage means (201);

data storage means (206) for storing musical tone generation data necessary for generating a musical tone signal based on the modulation method;

arithmetic processing means (208, 209) for processing data;

program execution means (101) for executing the performance data processing program and the sound source processing program stored in said program storage means (201) while controlling said address control means (205), said data storage means (206), and said arithmetic processing means (208, 209), for normally executing the performance data processing program to control musical tone generation data on said data storage means (206), for executing the sound source processing program at predetermined time intervals, for executing the performance data processing program again upon completion of the sound source processing program, and for generating a musical tone signal by the modulation method on the basis of the musical tone generation data on said data storage means (206) upon execution of said sound source processing program; and

musical tone signal output means (107, 108) for holding the musical tone signal obtained when said program execution means (101) executes the sound source processing program, and outputting the held musical tone signal at predetermined output time intervals.

14. An apparatus according to claim 13, characterized in that said program execution means (101) includes interrupt control means (203) for generating an interrupt signal at the predetermined time intervals, and

said program execution means (101) interrupts the performance data processing program at a generation timing of the interrupt signal from said interrupt control means (203) during execution of the performance data processing program, transits the control to the sound source processing program to execute the sound source processing program, cancels the interruption upon completion of the sound source processing program, and restarts execution of the performance data processing program.

15. An apparatus according to claim 13, characterized in that the sound source processing program based on the modulation method has a processing architecture in which algorithm processing for determining the input/output relationship of the operator processing operations is selected from a plurality of algorithm processing operations, and is executed at a time after or before execution of at least one operator processing as a modulation processing unit at a time.

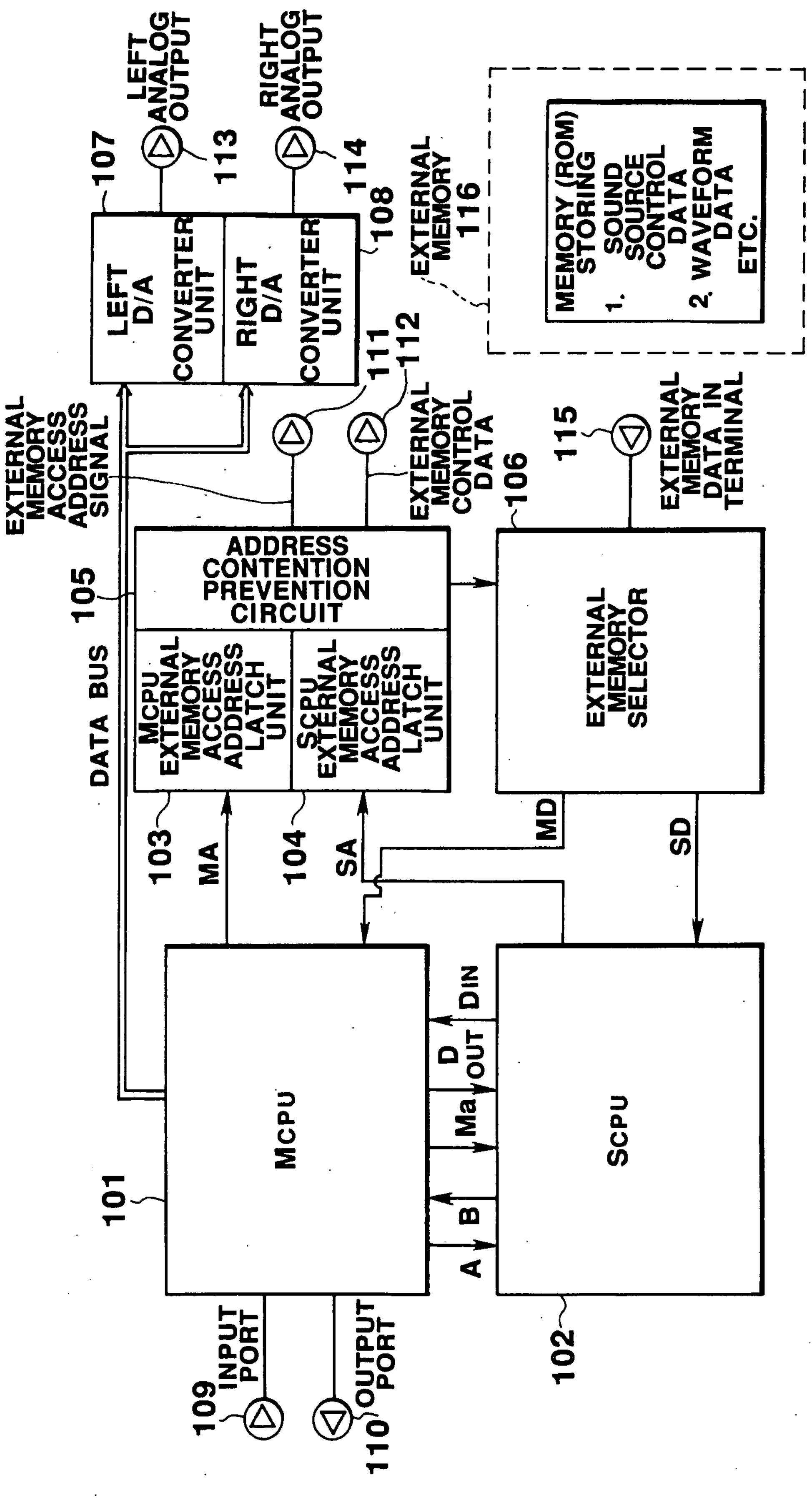

101
MCPU
102
SCPU
103
MCPU EXTERNAL MEMORY ACCESS ADDRESS LATCH UNIT
104
SCPU EXTERNAL MEMORY ACCESS ADDRESS LATCH UNIT
105
ADDRESS CONTENTION PREVENTION CIRCUIT
106
EXTERNAL MEMORY SELECTOR
107
LEFT D/A CONVERTER UNIT
108
RIGHT D/A CONVERTER UNIT
109
INPUT PORT
110
OUTPUT PORT
111
112
113
LEFT ANALOG OUTPUT
114
RIGHT ANALOG OUTPUT
115
EXTERNAL MEMORY DATA IN TERMINAL
116
EXTERNAL MEMORY
MEMORY (ROM) STORING
1. SOUND SOURCE CONTROL DATA
2. WAVEFORM DATA
ETC.
EXTERNAL MEMORY ACCESS ADDRESS SIGNAL
EXTERNAL MEMORY CONTROL DATA
DATA BUS
MA
SA
MD
SD
A
B
Ma
D OUT
DIN

FIG.1

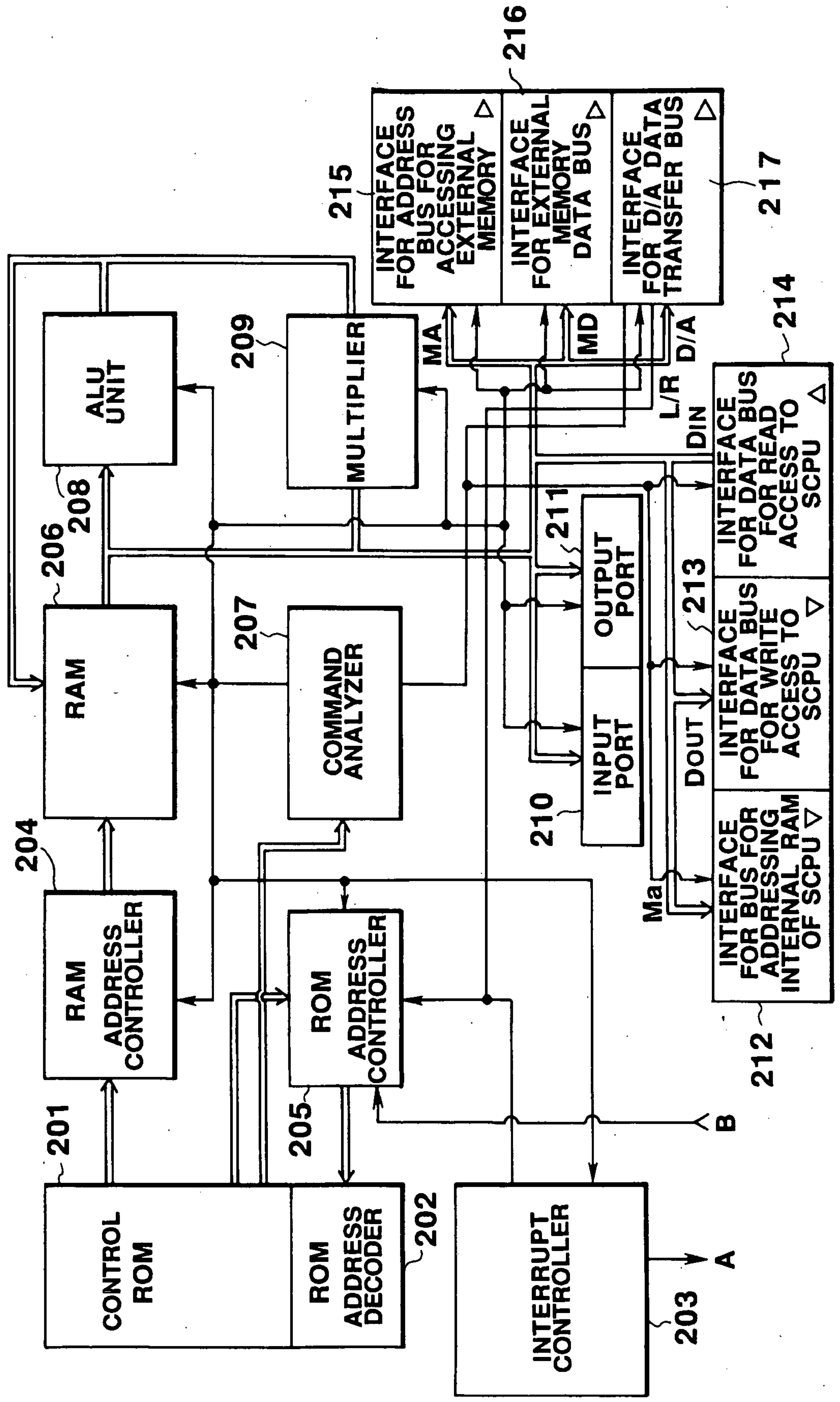

CONTROL ROM
201
ROM ADDRESS DECODER
202
INTERRUPT CONTROLLER
203
A
B
RAM ADDRESS CONTROLLER
204
ROM ADDRESS CONTROLLER
205
RAM
206
COMMAND ANALYZER
207
ALU UNIT
208
MULTIPLIER
209
INPUT PORT
210
OUTPUT PORT
211
MA
MD
D/A
L/R
DIN
DOUT
Ma
INTERFACE FOR ADDRESS BUS FOR ACCESSING EXTERNAL MEMORY
215
INTERFACE FOR EXTERNAL MEMORY DATA BUS
216
INTERFACE FOR D/A DATA TRANSFER BUS
217
INTERFACE FOR DATA BUS FOR READ ACCESS TO SCPU
214
INTERFACE FOR DATA BUS FOR WRITE ACCESS TO SCPU
213
INTERFACE FOR BUS FOR ADDRESSING INTERNAL RAM OF SCPU
212

FIG. 2

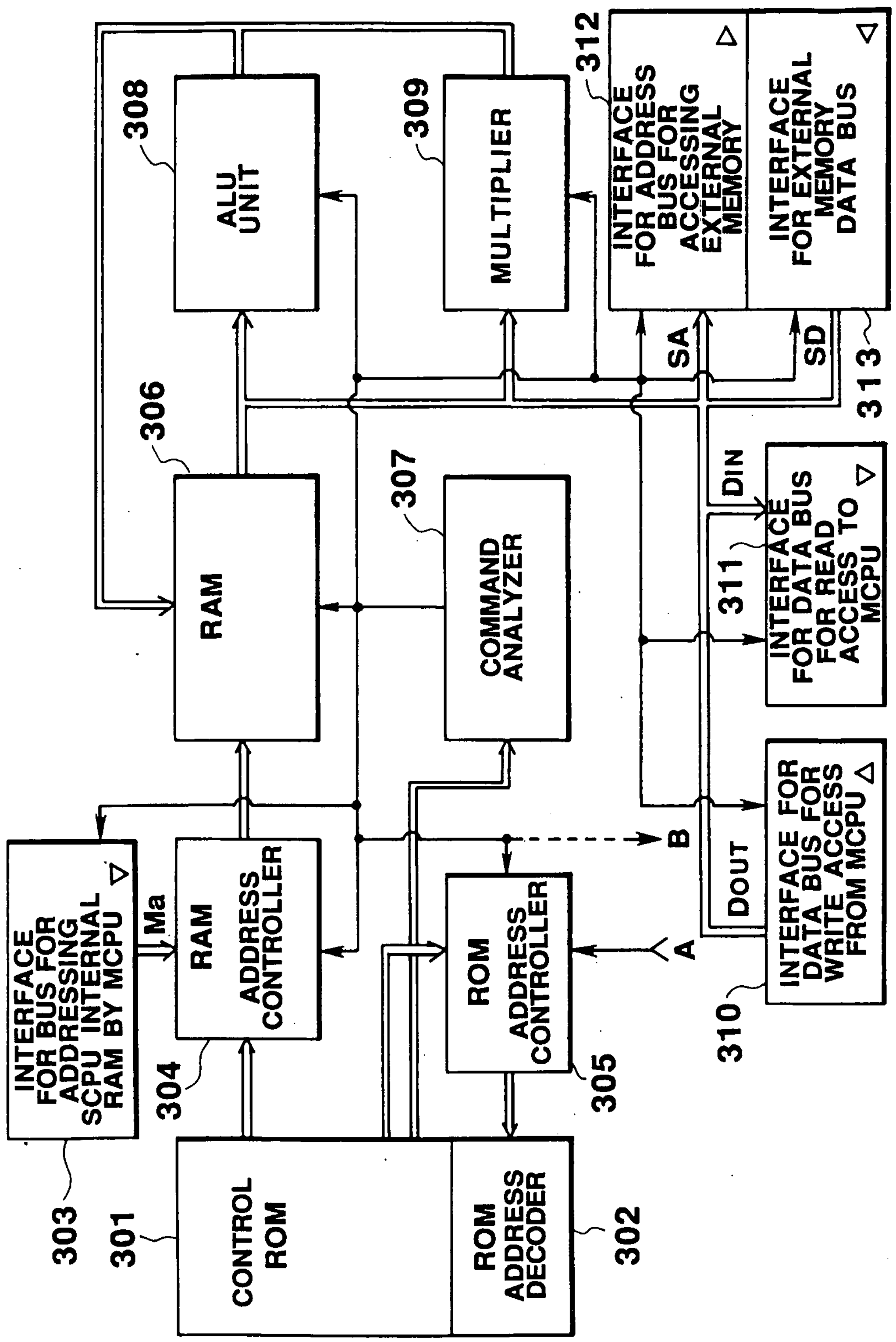

CONTROL ROM
301
ROM ADDRESS DECODER
302
INTERFACE FOR BUS FOR ADDRESSING SCPU INTERNAL RAM BY MCPU
303
Ma
RAM ADDRESS CONTROLLER
304
ROM ADDRESS CONTROLLER
305
A
B
RAM
306
COMMAND ANALYZER
307
ALU UNIT
308
MULTIPLIER
309
INTERFACE FOR DATA BUS FOR WRITE ACCESS FROM MCPU
310
DOUT
INTERFACE FOR DATA BUS FOR READ ACCESS TO MCPU
311
DIN
INTERFACE FOR ADDRESS BUS FOR ACCESSING EXTERNAL MEMORY
312
SA
INTERFACE FOR EXTERNAL MEMORY DATA BUS
313
SD

FIG. 3

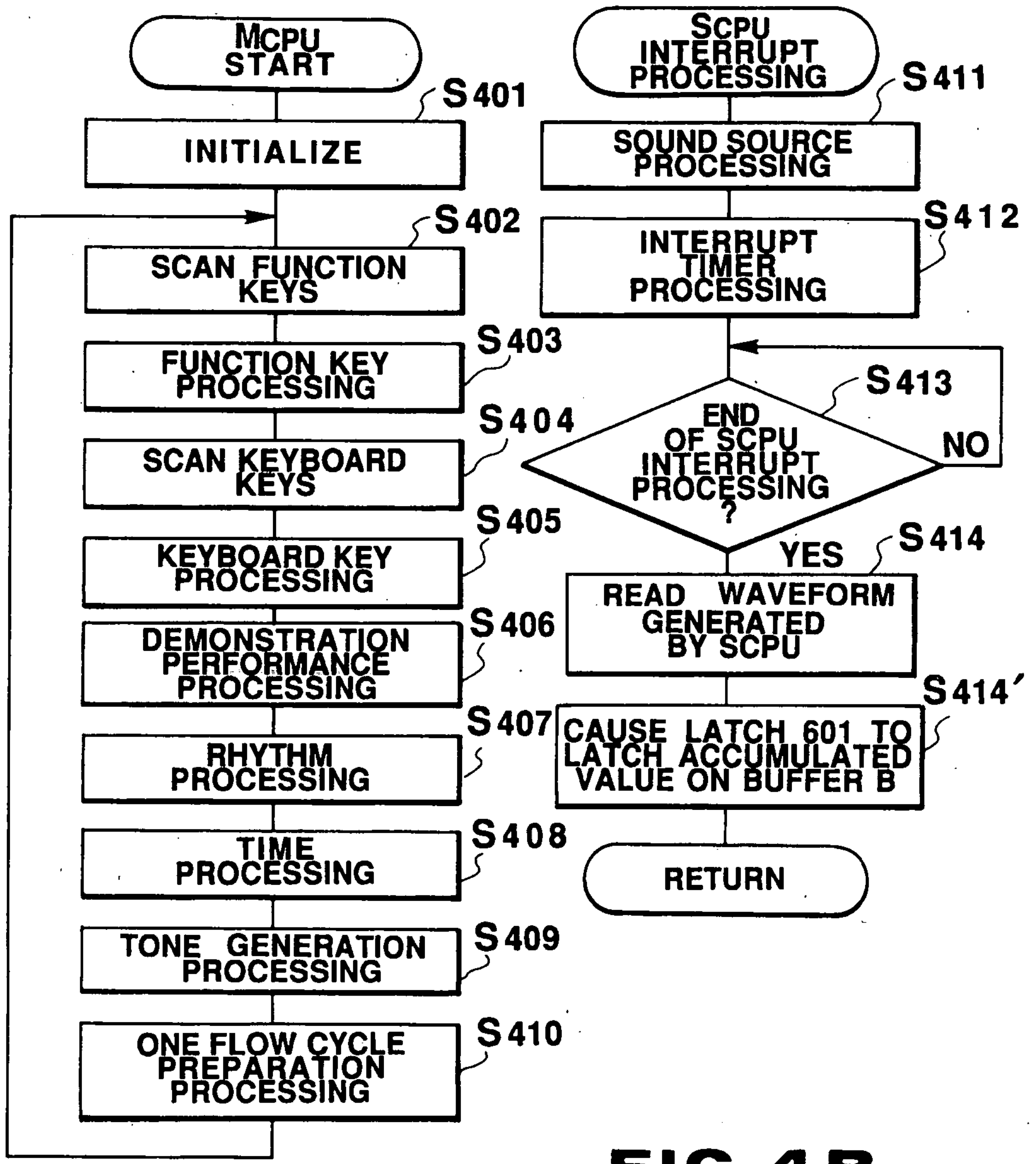

MCPU START
S401
INITIALIZE
S402
SCAN FUNCTION KEYS
S403
FUNCTION KEY PROCESSING
S404
SCAN KEYBOARD KEYS
S405
KEYBOARD KEY PROCESSING
S406
DEMONSTRATION PERFORMANCE PROCESSING
S407
RHYTHM PROCESSING
S408
TIME PROCESSING
S409
TONE GENERATION PROCESSING
S410
ONE FLOW CYCLE PREPARATION PROCESSING
SCPU INTERRUPT PROCESSING
S411
SOUND SOURCE PROCESSING
S412
INTERRUPT TIMER PROCESSING
S413
END OF SCPU INTERRUPT PROCESSING ?
NO
YES
S414
READ WAVEFORM GENERATED BY SCPU
S414′
CAUSE LATCH 601 TO LATCH ACCUMULATED VALUE ON BUFFER B
RETURN

FIG.4B

FIG.4A

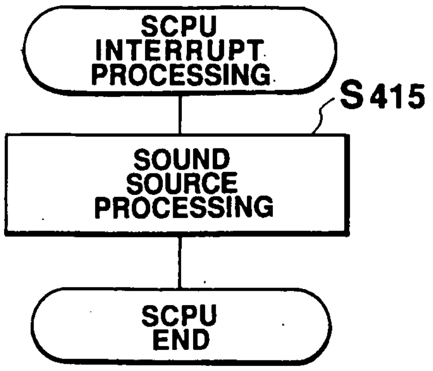

SCPU INTERRUPT PROCESSING
S415
SOUND SOURCE PROCESSING
SCPU END

FIG.4C

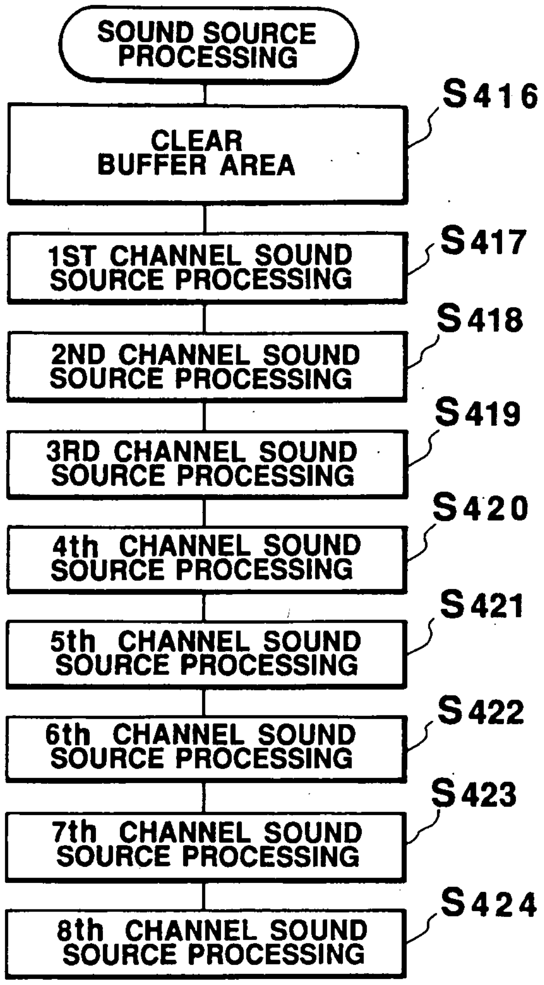

SOUND SOURCE PROCESSING
S416
CLEAR BUFFER AREA
S417
1ST CHANNEL SOUND SOURCE PROCESSING
S418
2ND CHANNEL SOUND SOURCE PROCESSING
S419
3RD CHANNEL SOUND SOURCE PROCESSING
S420
4th CHANNEL SOUND SOURCE PROCESSING
S421
5th CHANNEL SOUND SOURCE PROCESSING
S422
6th CHANNEL SOUND SOURCE PROCESSING
S423
7th CHANNEL SOUND SOURCE PROCESSING
S424
8th CHANNEL SOUND SOURCE PROCESSING

FIG.4D

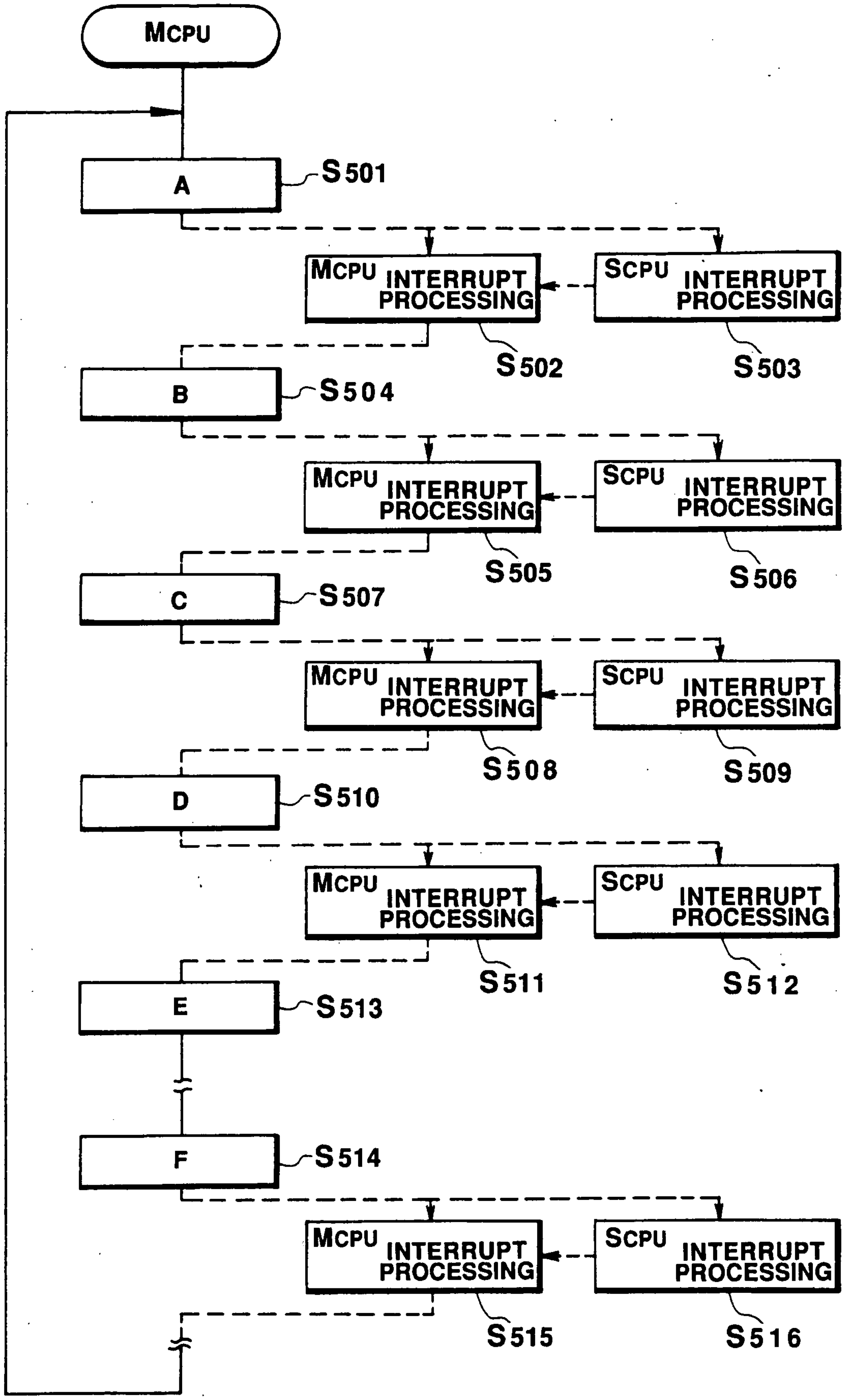

MCPU
A
S501
MCPU INTERRUPT PROCESSING
SCPU INTERRUPT PROCESSING
S502
S503
B
S504
MCPU INTERRUPT PROCESSING
SCPU INTERRUPT PROCESSING
S505
S506
C
S507
MCPU INTERRUPT PROCESSING
SCPU INTERRUPT PROCESSING
S508
S509
D
S510
MCPU INTERRUPT PROCESSING
SCPU INTERRUPT PROCESSING
S511
S512
E
S513
F
S514
MCPU INTERRUPT PROCESSING
SCPU INTERRUPT PROCESSING
S515
S516

FIG.5

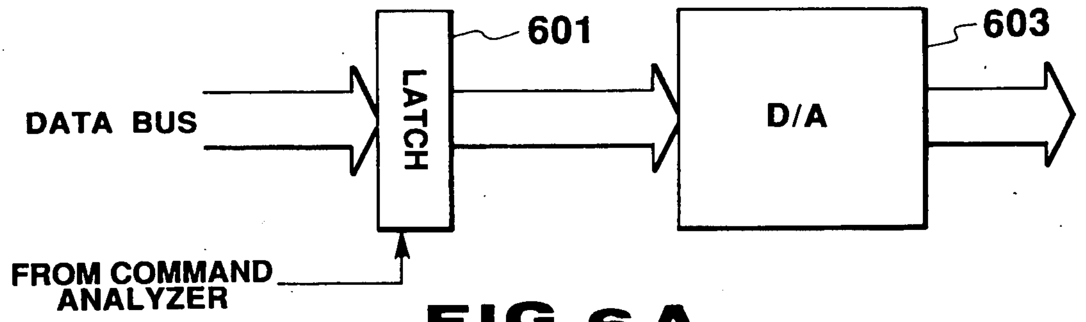

DATA BUS
LATCH
601
D/A
603
FROM COMMAND ANALYZER

FIG.6 A

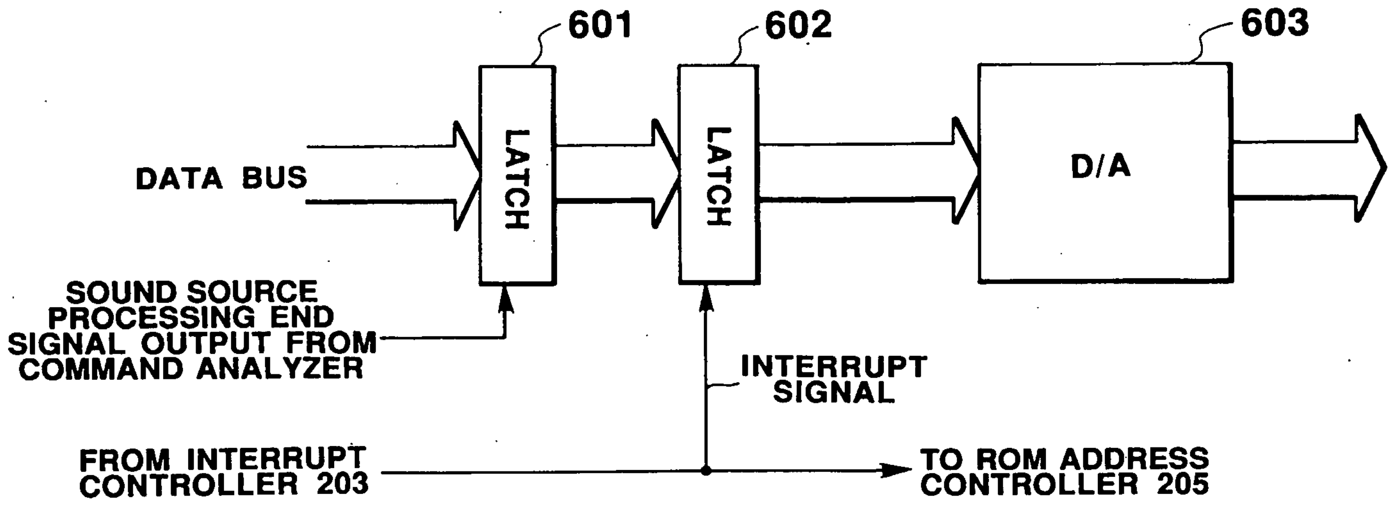

601
602
603
DATA BUS
LATCH
LATCH
D/A
SOUND SOURCE PROCESSING END SIGNAL OUTPUT FROM COMMAND ANALYZER
INTERRUPT SIGNAL
FROM INTERRUPT CONTROLLER 203
TO ROM ADDRESS CONTROLLER 205

FIG.6 B

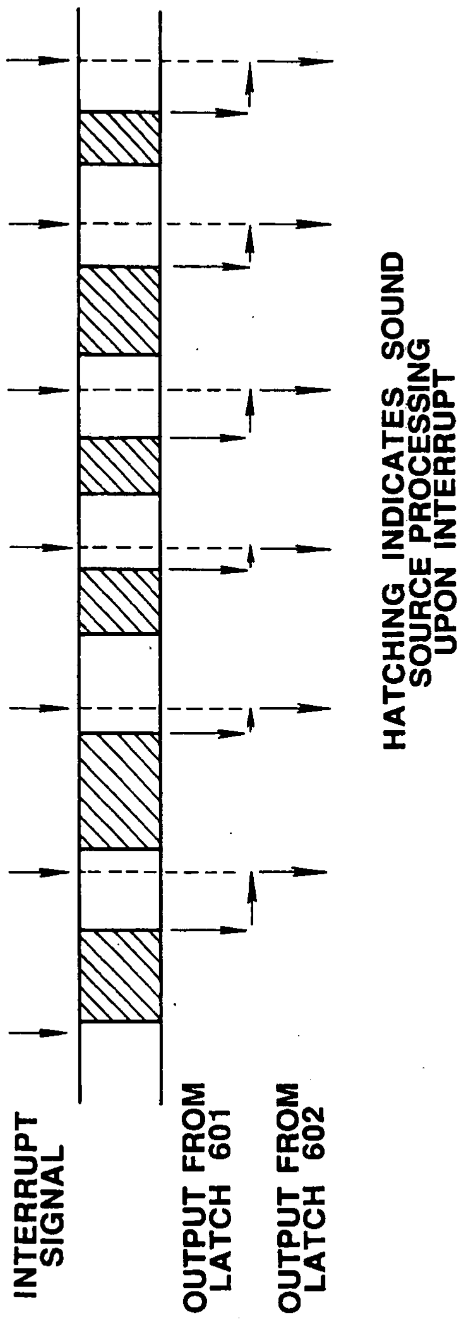

INTERRUPT SIGNAL
OUTPUT FROM LATCH 601
OUTPUT FROM LATCH 602
HATCHING INDICATES SOUND SOURCE PROCESSING UPON INTERRUPT

FIG.7

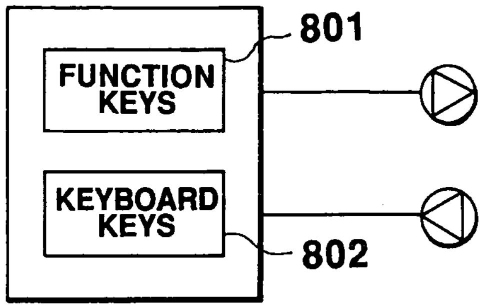

801
FUNCTION KEYS
KEYBOARD KEYS
802

FIG.8 A

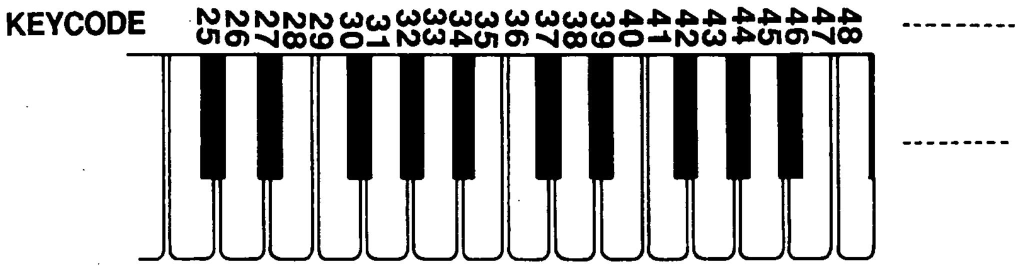

KEYCODE
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48

FIG.8 B

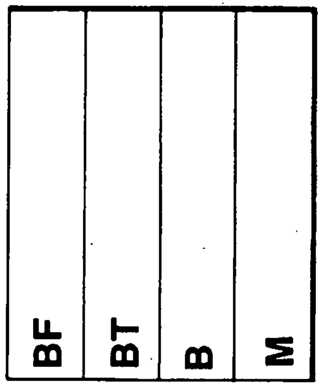


FIG. 12

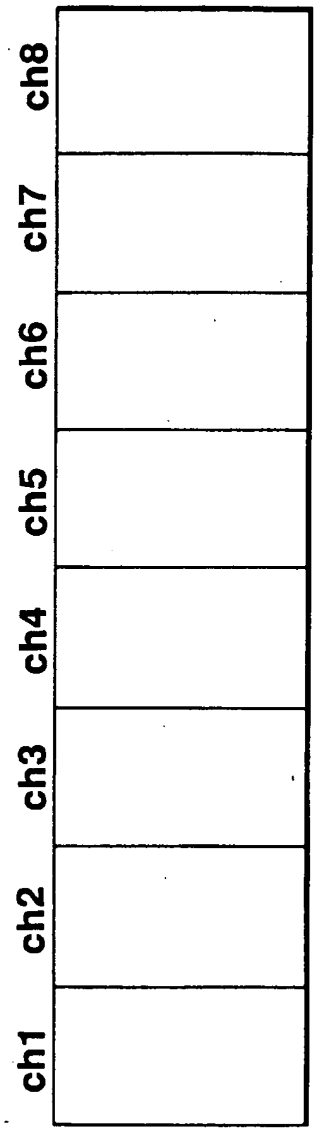


FIG. 9

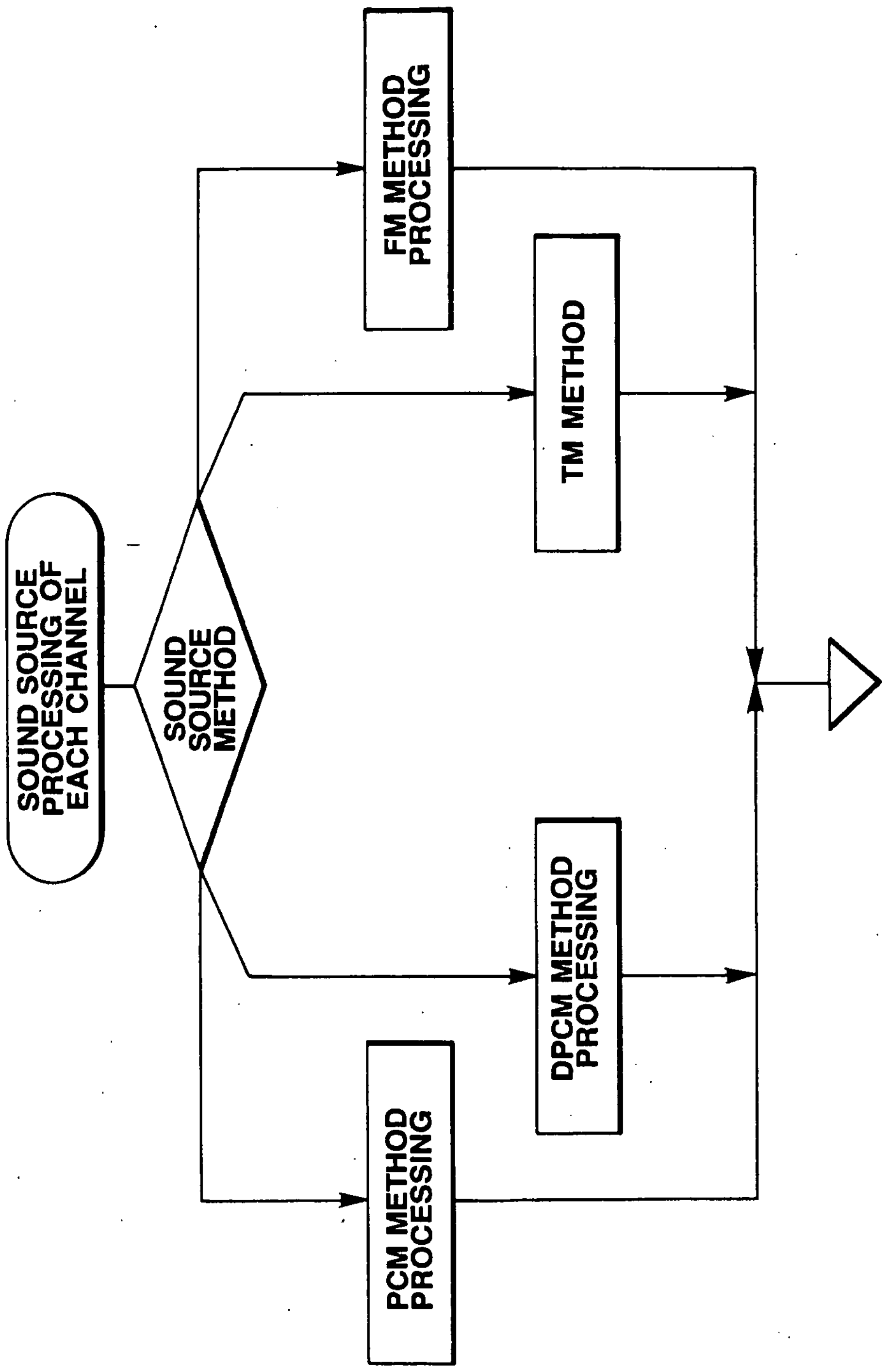

SOUND SOURCE PROCESSING OF EACH CHANNEL
SOUND SOURCE METHOD
FM METHOD PROCESSING
TM METHOD
DPCM METHOD PROCESSING
PCM METHOD PROCESSING

FIG.10

PCM		DPCM		FM		TM	
G	SOUND SOURCE METHOD NO.	G	SOUND SOURCE METHOD NO.	G	SOUND SOURCE METHOD NO.	G	SOUND SOURCE METHOD NO.
A_I	CURRENT ADDRESS INTEGRAL PART	A_I	CURRENT ADDRESS INTEGRAL PART	A_1	CURRENT ADDRESS (OP1)	A_1	CURRENT ADDRESS (OP1)
A_F	CURRENT ADDRESS DECIMAL PART	A_F	CURRENT ADDRESS DECIMAL PART	A_2	CURRENT ADDRESS (OP2)	A_2	CURRENT ADDRESS (OP2)
A_E	END ADDRESS INTEGRAL PART	A_E	END ADDRESS INTEGRAL PART				
A_L	LOOP ADDRESS INTEGRAL PART	A_L	LOOP ADDRESS INTEGRAL PART				
P_F	PITCH DATA DECIMAL PART	P_F	PITCH DATA DECIMAL PART	P_1	PITCH DATA (OP1)	P_1	PITCH DATA (OP1)
P_I	PITCH DATA INTEGRAL PART	P_I	PITCH DATA INTEGRAL PART	P_2	PITCH DATA (OP2)	P_2	PITCH DATA (OP2)
X_P	IMMEDIATELY PRECEDING SAMPLE DATA	X_P	IMMEDIATELY PRECEDING SAMPLE DATA				
X_N	NEXT SAMPLE DATA	OLD A_I	CURRENT ADDRESS INTEGRAL PART BEFORE CHANGE				
D	DIFFERENCE BETWEEN ADJACENT SAMPLE DATA	D	DIFFERENCE BETWEEN ADJACENT SAMPLE DATA				
E	ENVELOPE VALUE	E	ENVELOPE VALUE	E_1	ENVELOPE (OP1)	E_1	ENVELOPE (OP1)
		X_{PL}	SAMPLE DATA OF AL	E_2	ENVELOPE (OP2)	E_2	ENVELOPE (OP2)
				M_{L2}	MODULATION LEVEL (OP2)	M_{L2}	MODULATION LEVEL (OP2)
				M_{O2}	MODULATION OUTPUT (OP2)	M_{O2}	MODULATION OUTPUT (OP2)
				F_{L2}	FEEDBACK LEVEL (OP2)	F_{L2}	FEEDBACK LEVEL (OP2)
				F_{O2}	FEEDBACK OUTPUT (OP2)	F_{O2}	FEEDBACK OUTPUT (OP2)
O	OUTPUT	O	OUTPUT	O_1	OP1 OUTPUT	O_1	OP1 OUTPUT
				O_2	OP2 OUTPUT	O_2	OP2 OUTPUT
C		C		C		C	

FIG.11

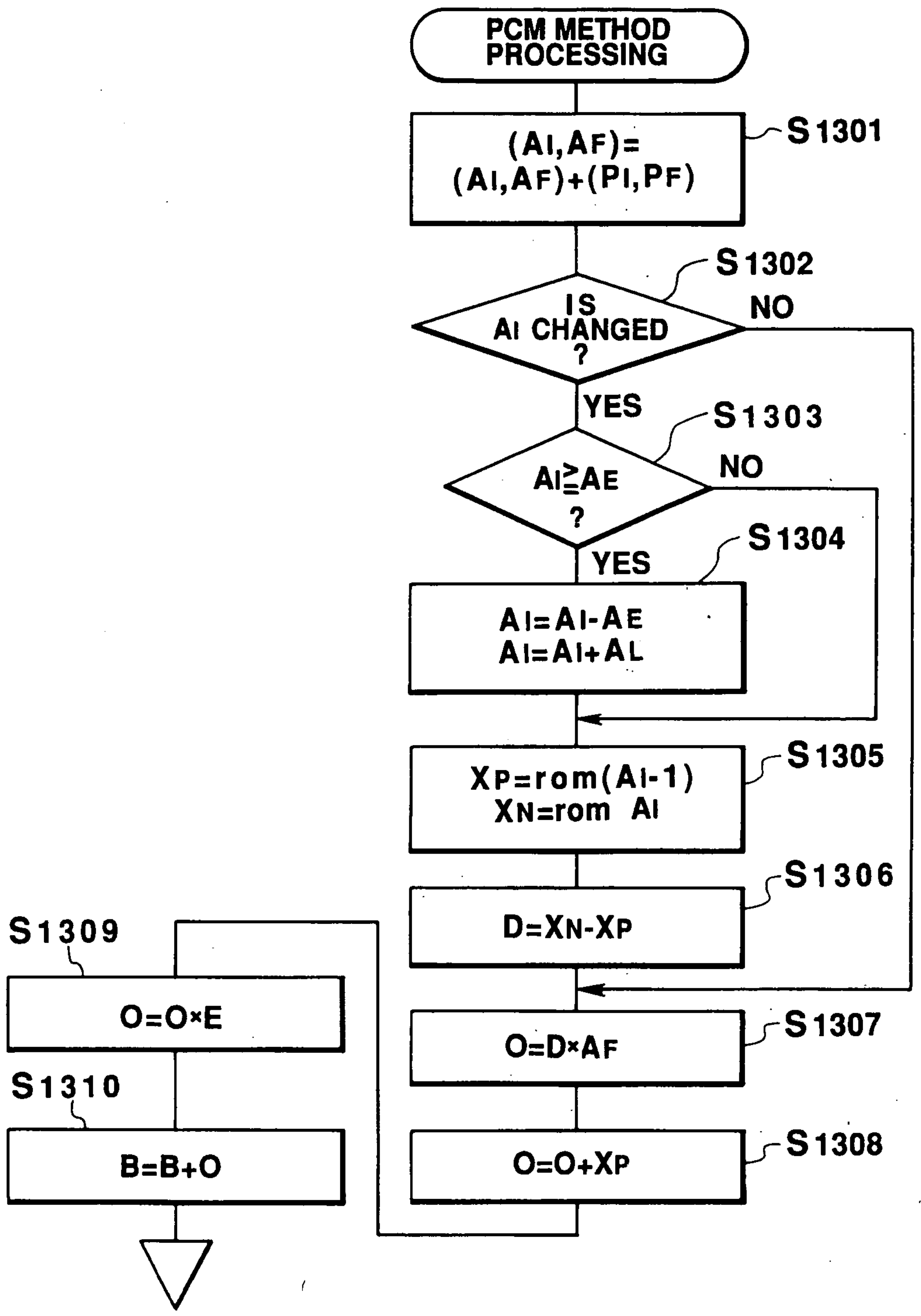

PCM METHOD PROCESSING
(AI,AF)=(AI,AF)+(PI,PF)
S1301
IS AI CHANGED ?
S1302
NO
YES
AI≧AE ?
S1303
NO
YES
AI=AI-AE
AI=AI+AL
S1304
XP=rom(AI-1)
XN=rom AI
S1305
D=XN-XP
S1306
O=D×AF
S1307
O=O+XP
S1308
O=O×E
S1309
B=B+O
S1310

FIG.13

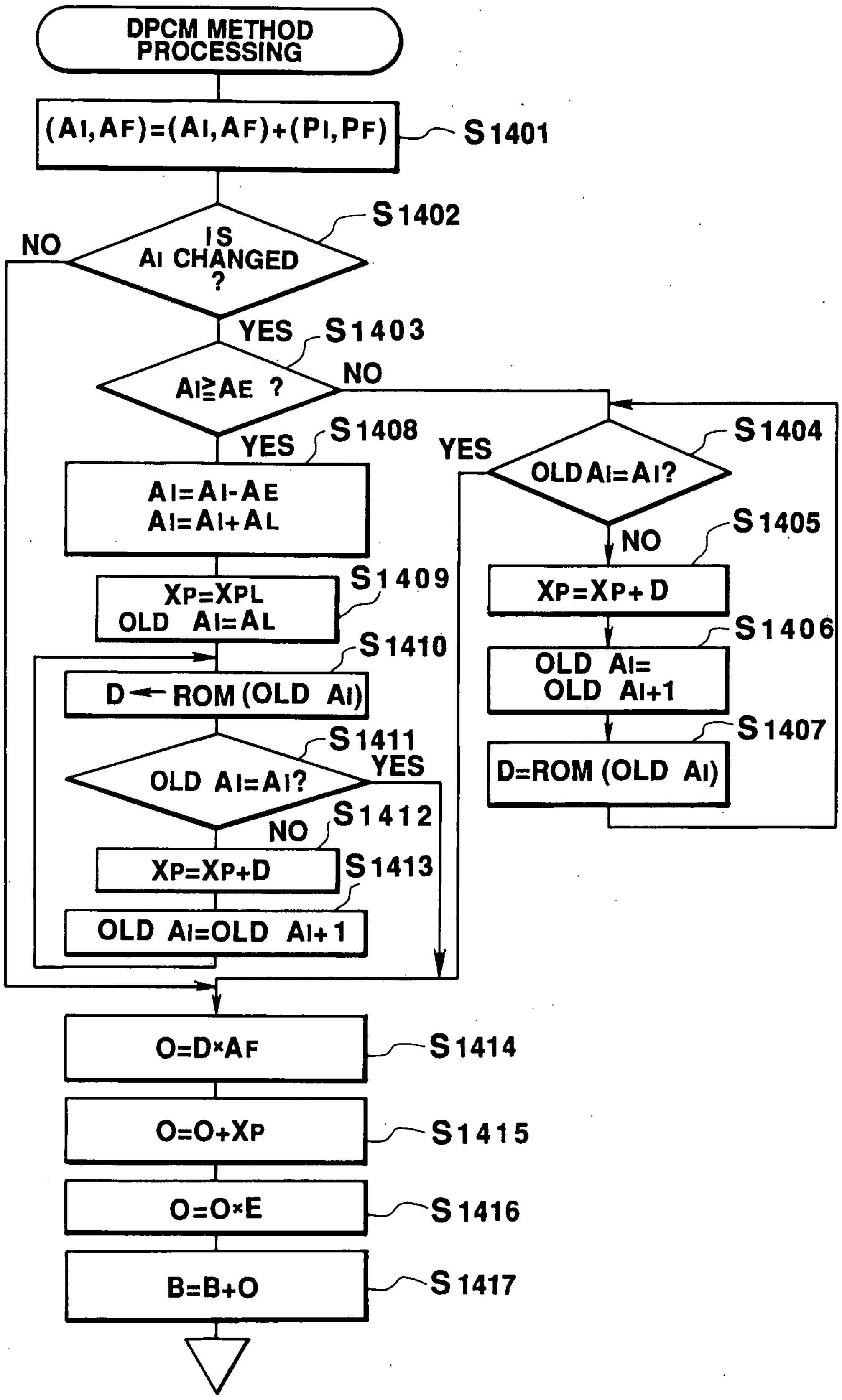

DPCM METHOD PROCESSING
(AI,AF)=(AI,AF)+(PI,PF)
S1401
S1402
IS AI CHANGED ?
NO
YES
S1403
AI≧AE ?
NO
YES
S1408
AI=AI-AE
AI=AI+AL
S1409
XP=XPL
OLD AI=AL
S1410
D←ROM (OLD AI)
S1411
OLD AI=AI?
YES
NO
S1412
XP=XP+D
S1413
OLD AI=OLD AI+1
S1404
OLD AI=AI?
YES
NO
S1405
XP=XP+D
S1406
OLD AI=
OLD AI+1
S1407
D=ROM (OLD AI)
O=D×AF
S1414
O=O+XP
S1415
O=O×E
S1416
B=B+O
S1417

FIG.14

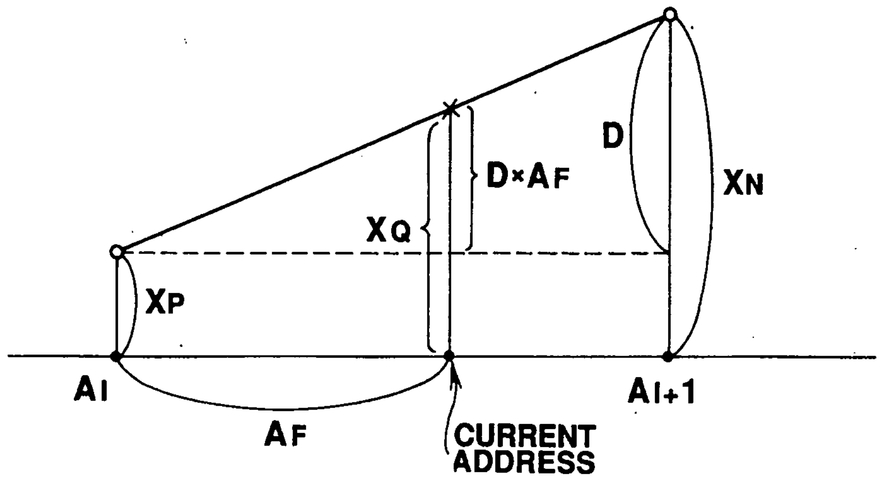


FIG.15A

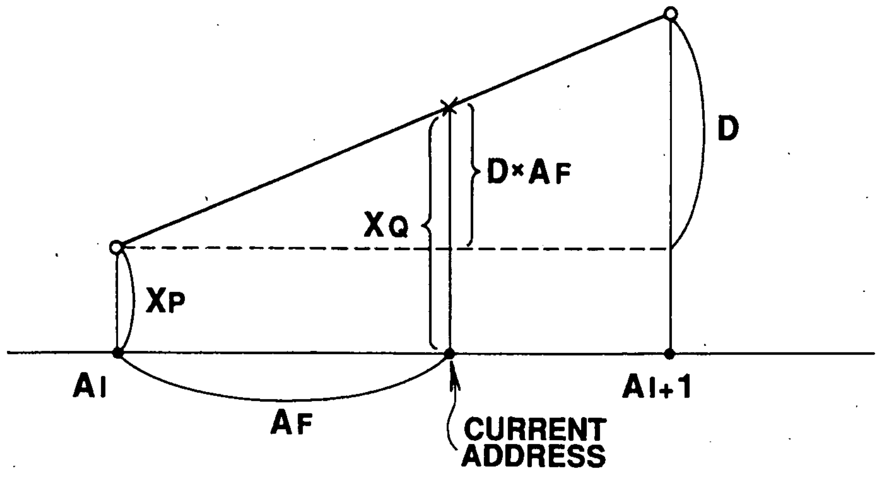


FIG.15B

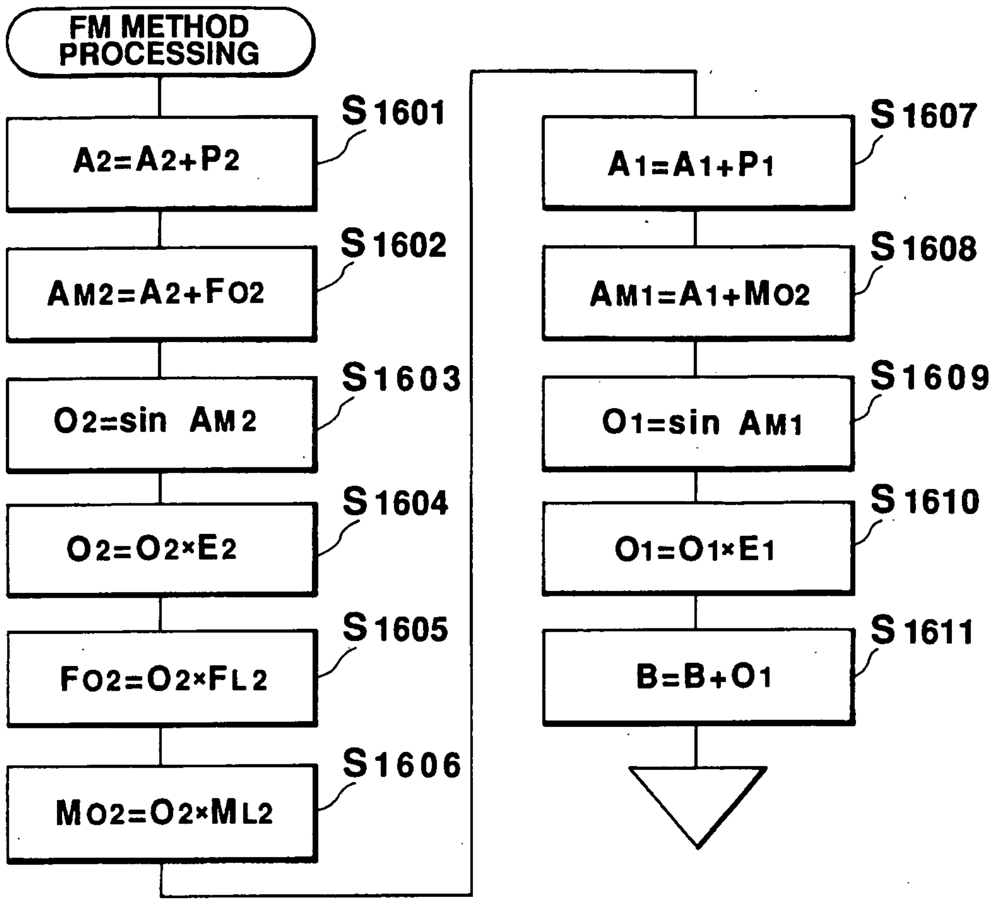


FIG.16A

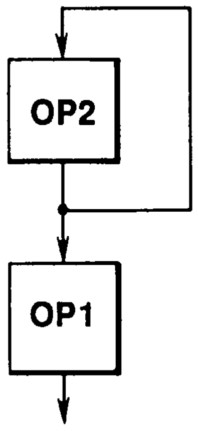


FIG.16B

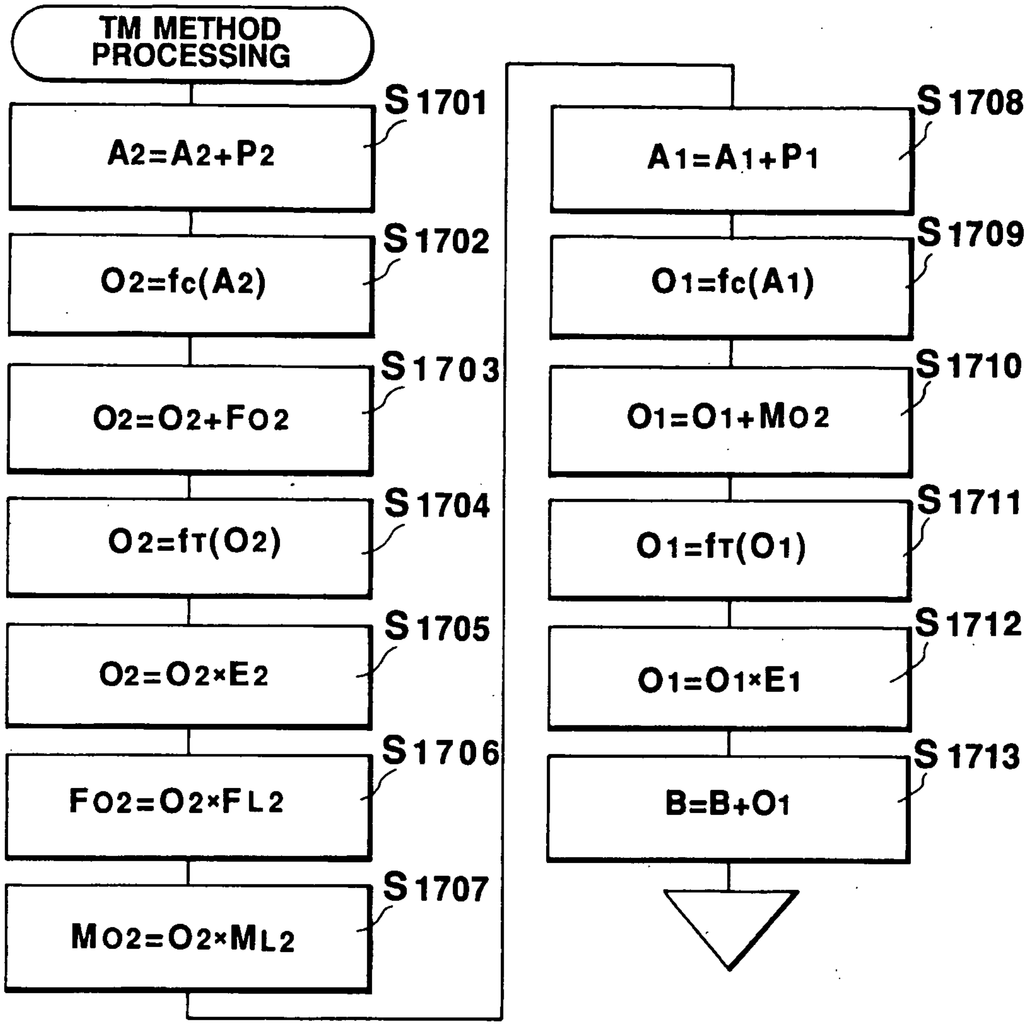


FIG.17 A

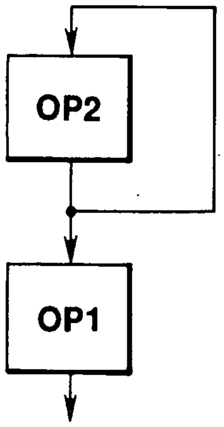
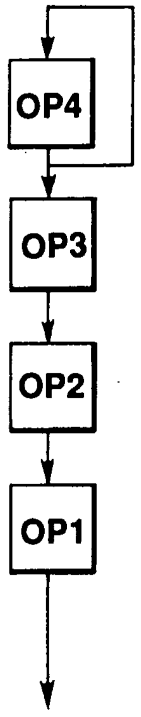
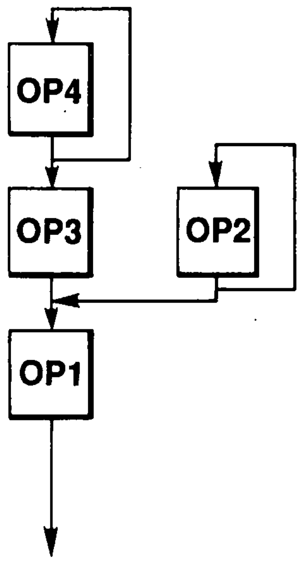


FIG.17 B





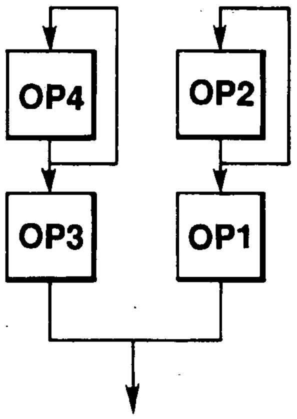


FIG.18A FIG.18B FIG.18C

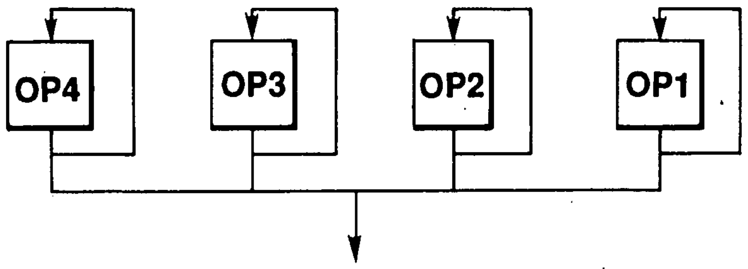


FIG.18D

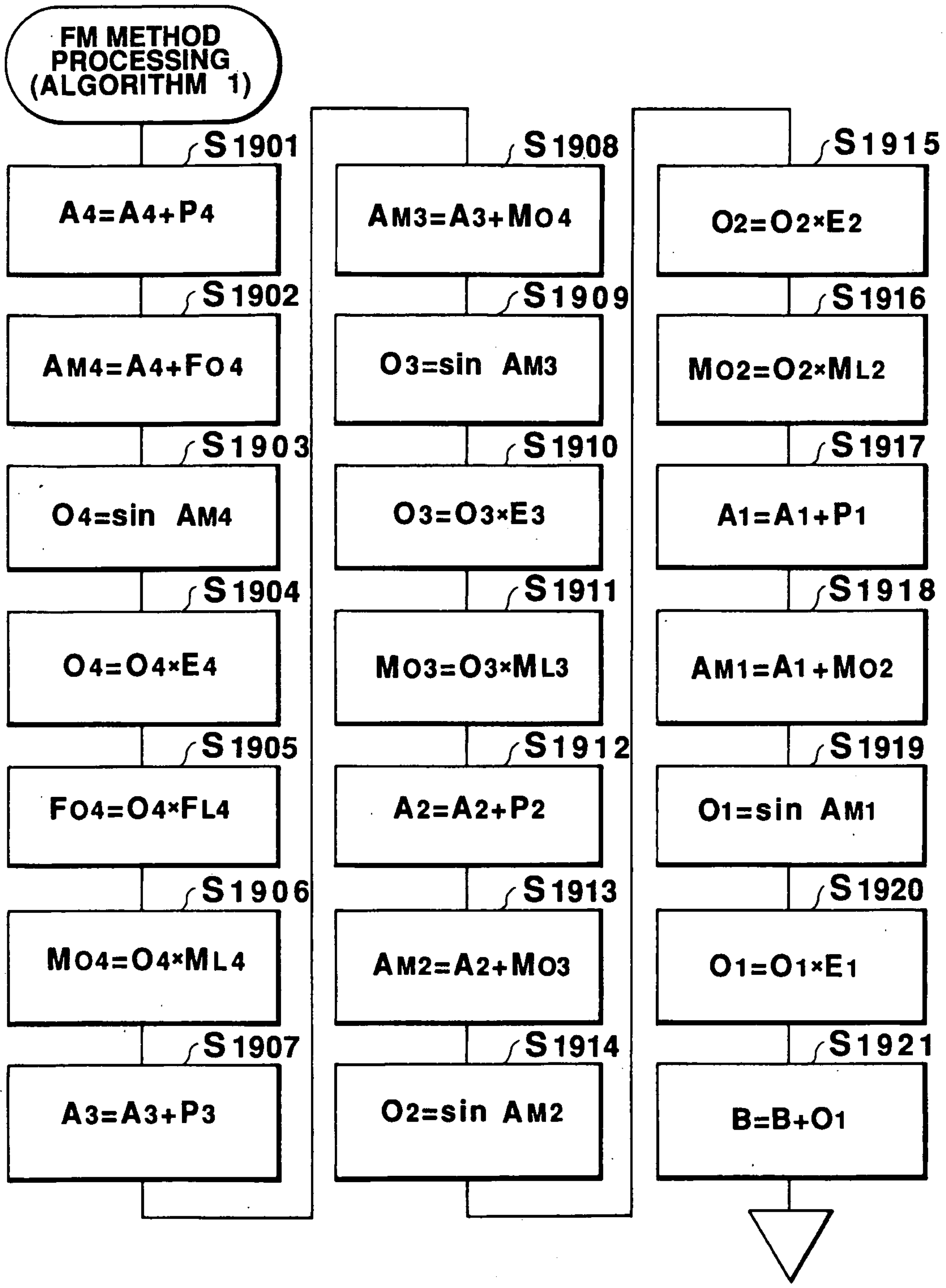

FM METHOD PROCESSING (ALGORITHM 1)
S1901
A4=A4+P4
S1902
AM4=A4+FO4
S1903
O4=sin AM4
S1904
O4=O4×E4
S1905
FO4=O4×FL4
S1906
MO4=O4×ML4
S1907
A3=A3+P3
S1908
AM3=A3+MO4
S1909
O3=sin AM3
S1910
O3=O3×E3
S1911
MO3=O3×ML3
S1912
A2=A2+P2
S1913
AM2=A2+MO3
S1914
O2=sin AM2
S1915
O2=O2×E2
S1916
MO2=O2×ML2
S1917
A1=A1+P1
S1918
AM1=A1+MO2
S1919
O1=sin AM1
S1920
O1=O1×E1
S1921
B=B+O1

FIG.19

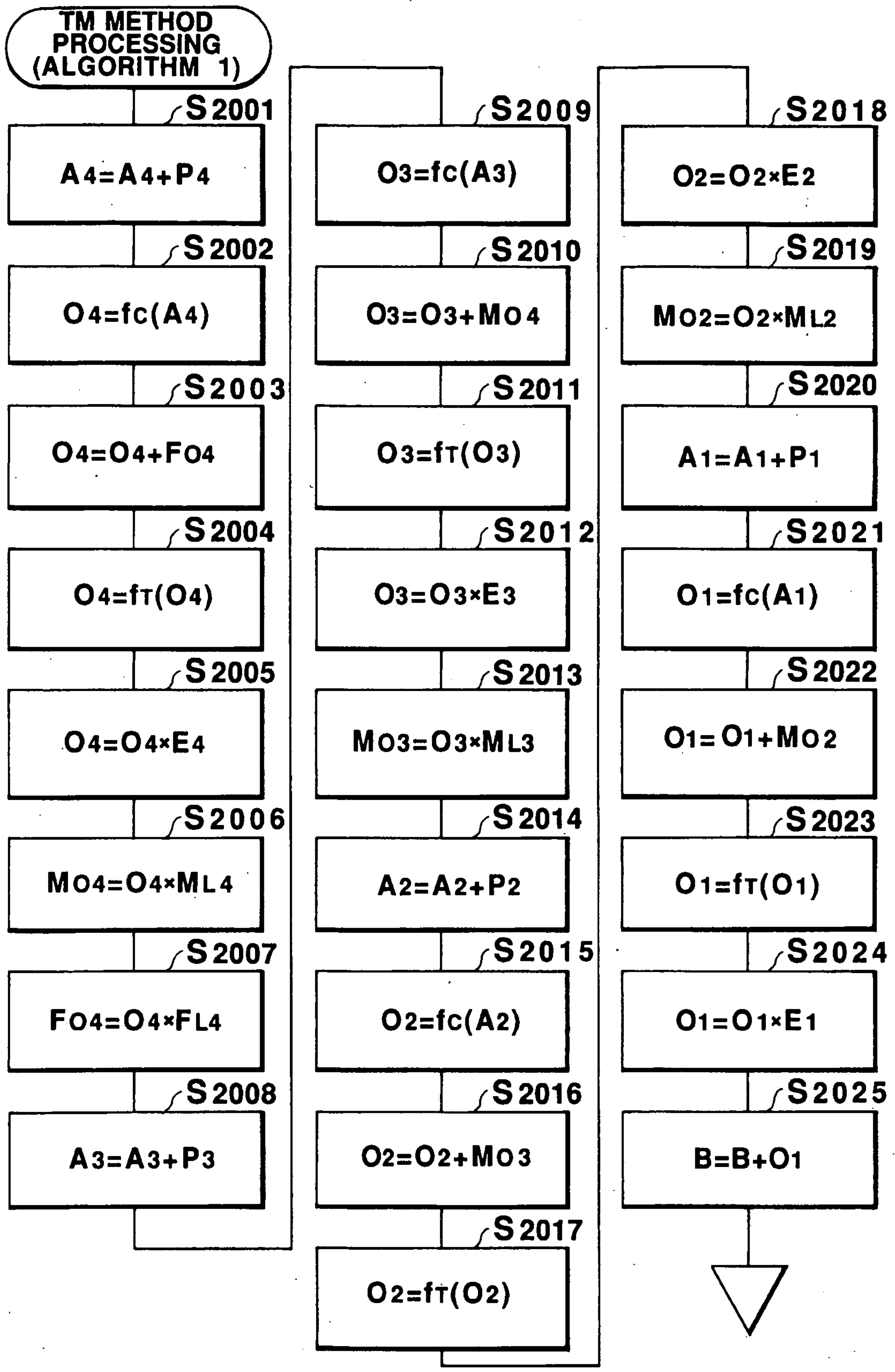

TM METHOD PROCESSING (ALGORITHM 1)
S2001
$A_4=A_4+P_4$
S2002
$O_4=f_C(A_4)$
S2003
$O_4=O_4+F_{O4}$
S2004
$O_4=f_T(O_4)$
S2005
$O_4=O_4\times E_4$
S2006
$M_{O4}=O_4\times M_{L4}$
S2007
$F_{O4}=O_4\times F_{L4}$
S2008
$A_3=A_3+P_3$
S2009
$O_3=f_C(A_3)$
S2010
$O_3=O_3+M_{O4}$
S2011
$O_3=f_T(O_3)$
S2012
$O_3=O_3\times E_3$
S2013
$M_{O3}=O_3\times M_{L3}$
S2014
$A_2=A_2+P_2$
S2015
$O_2=f_C(A_2)$
S2016
$O_2=O_2+M_{O3}$
S2017
$O_2=f_T(O_2)$
S2018
$O_2=O_2\times E_2$
S2019
$M_{O2}=O_2\times M_{L2}$
S2020
$A_1=A_1+P_1$
S2021
$O_1=f_C(A_1)$
S2022
$O_1=O_1+M_{O2}$
S2023
$O_1=f_T(O_1)$
S2024
$O_1=O_1\times E_1$
S2025
$B=B+O_1$

FIG.20

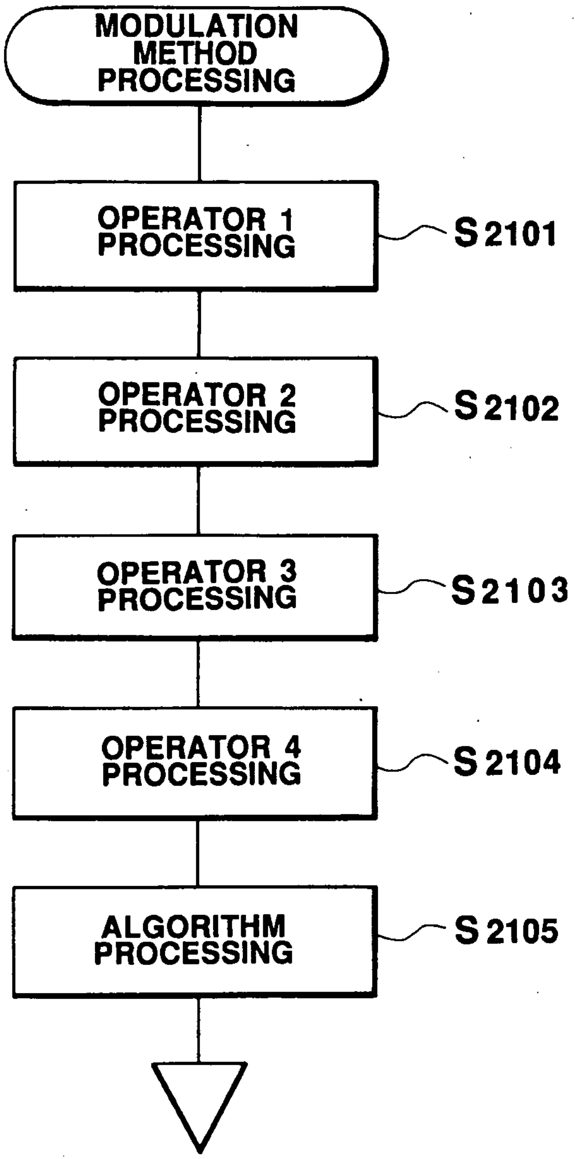

MODULATION METHOD PROCESSING
OPERATOR 1 PROCESSING
S2101
OPERATOR 2 PROCESSING
S2102
OPERATOR 3 PROCESSING
S2103
OPERATOR 4 PROCESSING
S2104
ALGORITHM PROCESSING
S2105

FIG.21

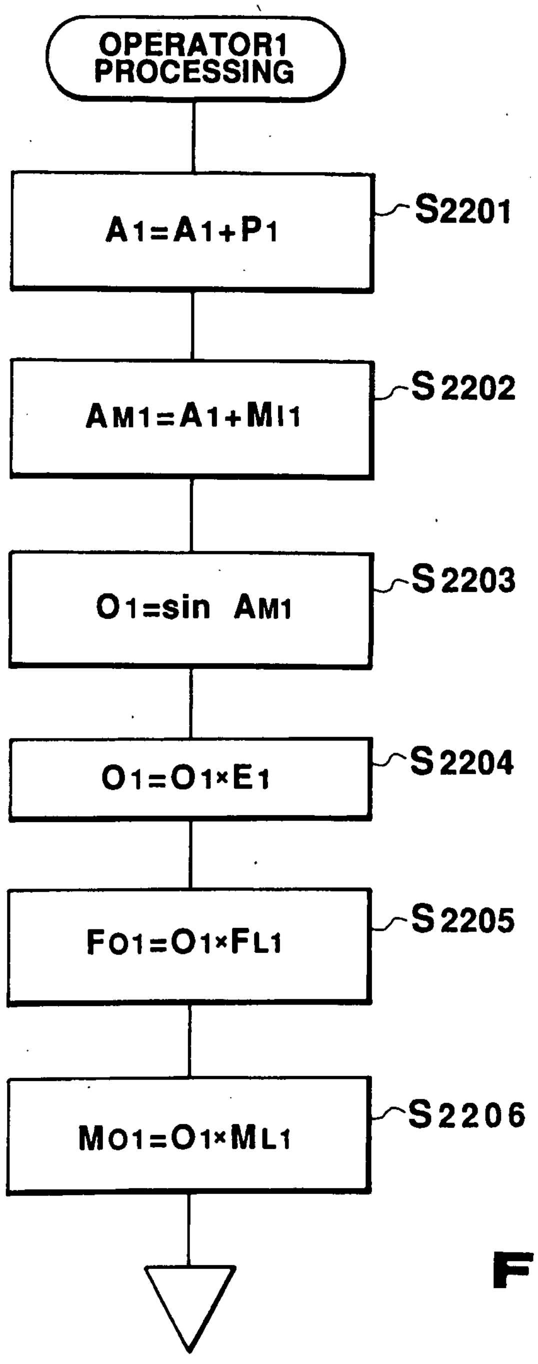


FIG.22A

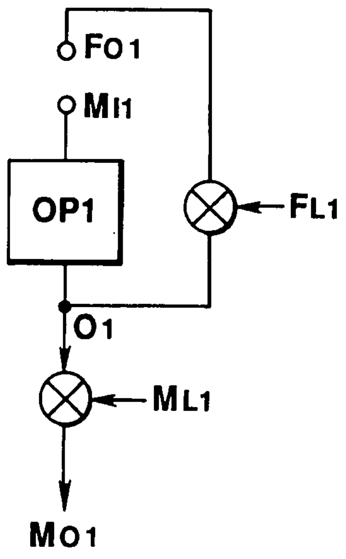


FIG.22B

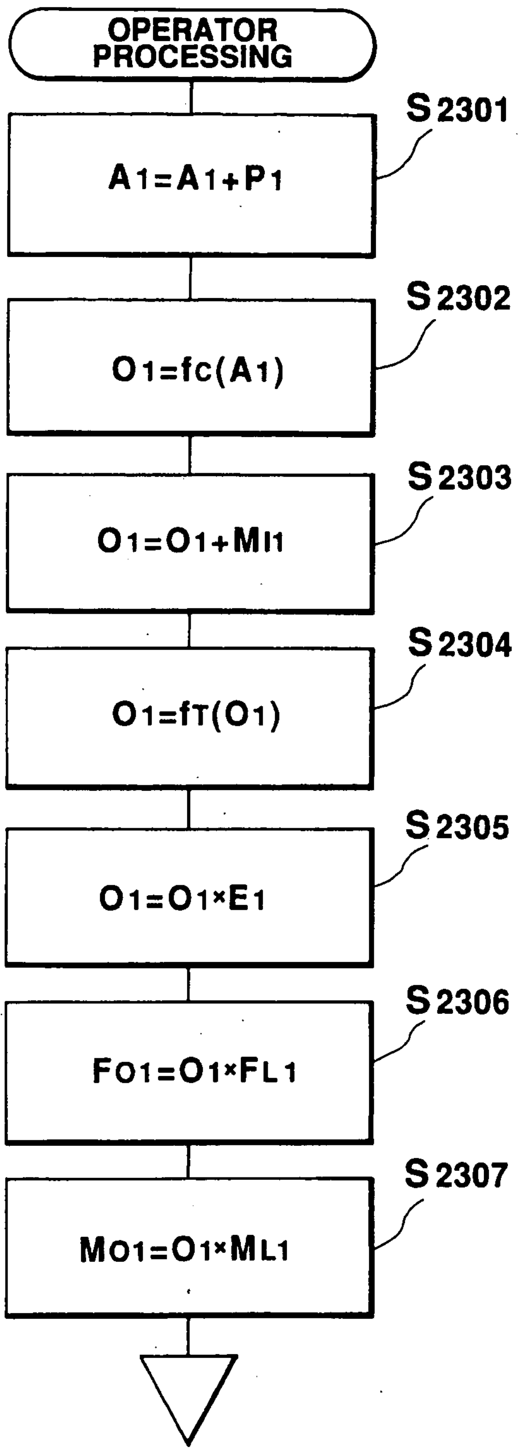


FIG.23A

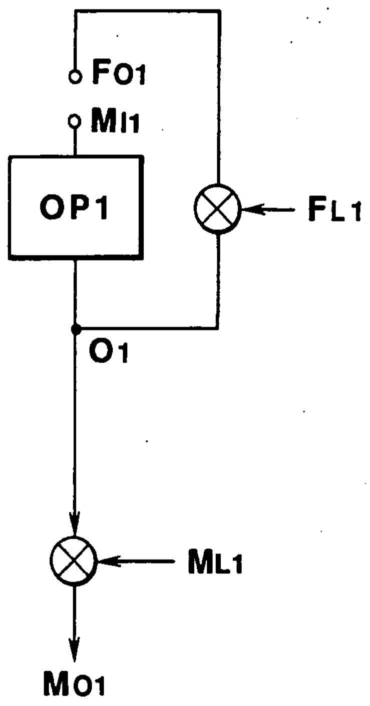


FIG.23B

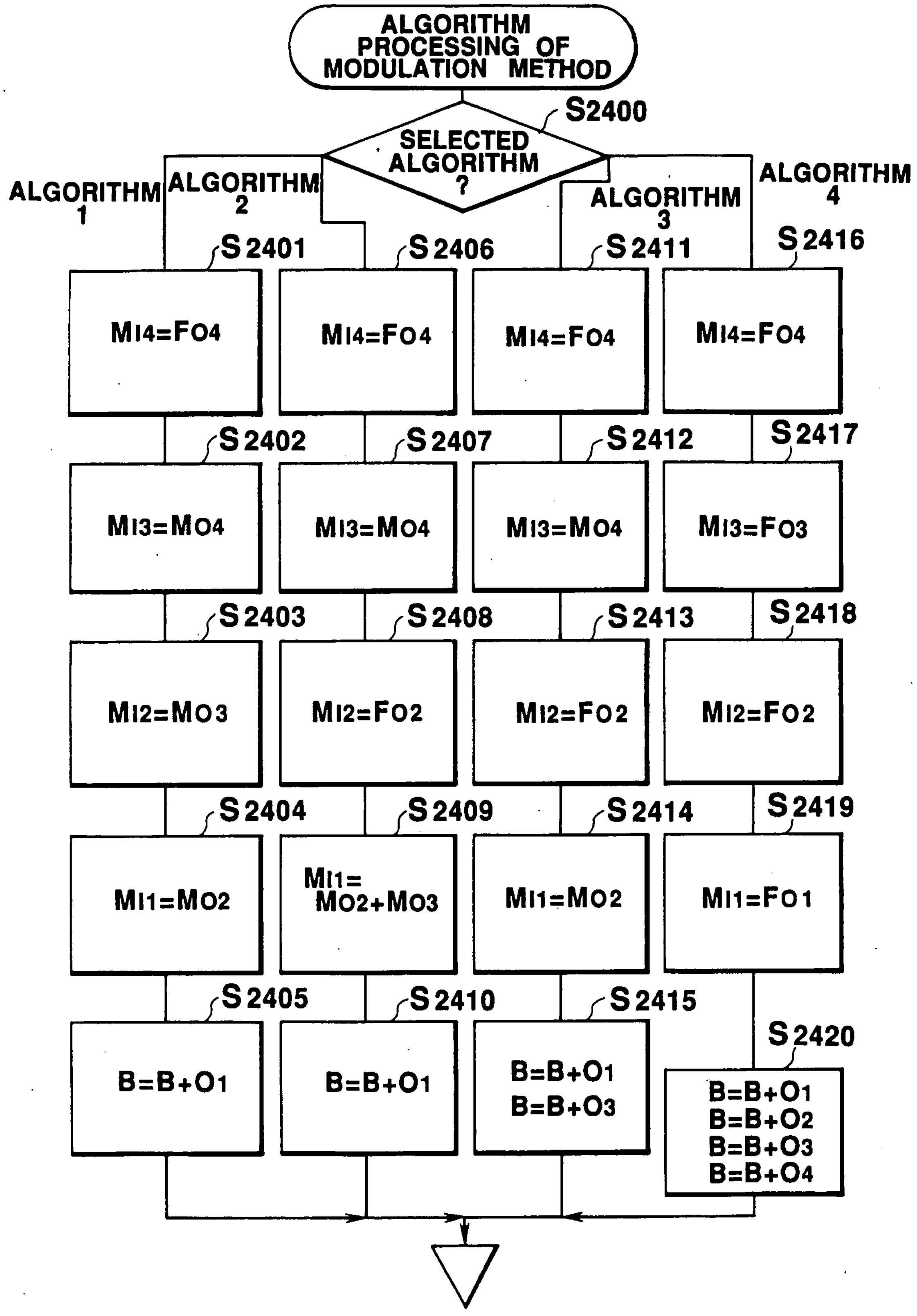

ALGORITHM PROCESSING OF MODULATION METHOD
S2400
SELECTED ALGORITHM ?
ALGORITHM 1
ALGORITHM 2
ALGORITHM 3
ALGORITHM 4
S2401
MI4=FO4
S2402
MI3=MO4
S2403
MI2=MO3
S2404
MI1=MO2
S2405
B=B+O1
S2406
MI4=FO4
S2407
MI3=MO4
S2408
MI2=FO2
S2409
MI1=MO2+MO3
S2410
B=B+O1
S2411
MI4=FO4
S2412
MI3=MO4
S2413
MI2=FO2
S2414
MI1=MO2
S2415
B=B+O1
B=B+O3
S2416
MI4=FO4
S2417
MI3=FO3
S2418
MI2=FO2
S2419
MI1=FO1
S2420
B=B+O1
B=B+O2
B=B+O3
B=B+O4

FIG.24

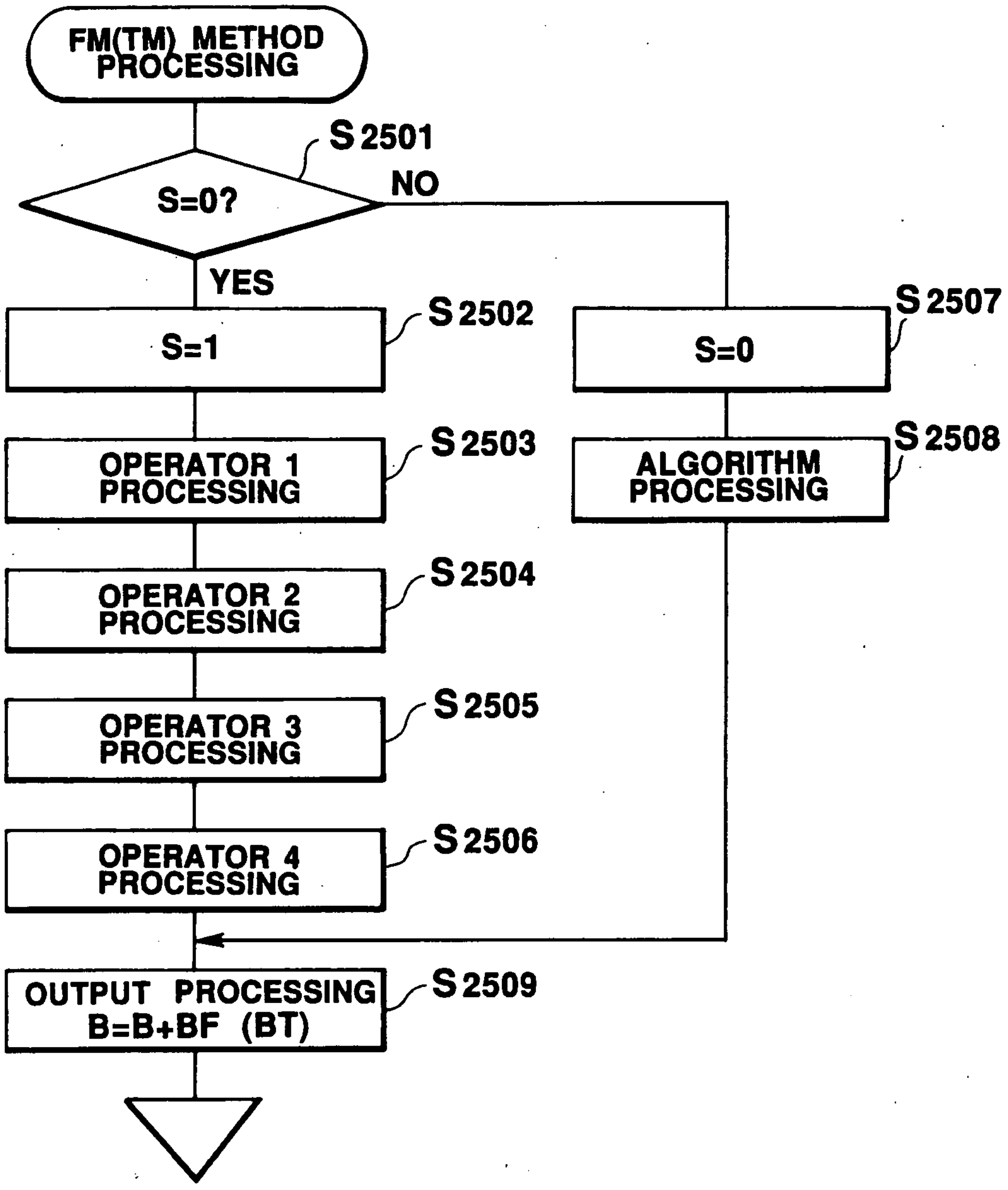

FM(TM) METHOD PROCESSING
S2501
S=0?
NO
YES
S2502
S=1
S2507
S=0
S2503
OPERATOR 1 PROCESSING
S2508
ALGORITHM PROCESSING
S2504
OPERATOR 2 PROCESSING
S2505
OPERATOR 3 PROCESSING
S2506
OPERATOR 4 PROCESSING
S2509
OUTPUT PROCESSING B=B+BF (BT)

FIG. 25

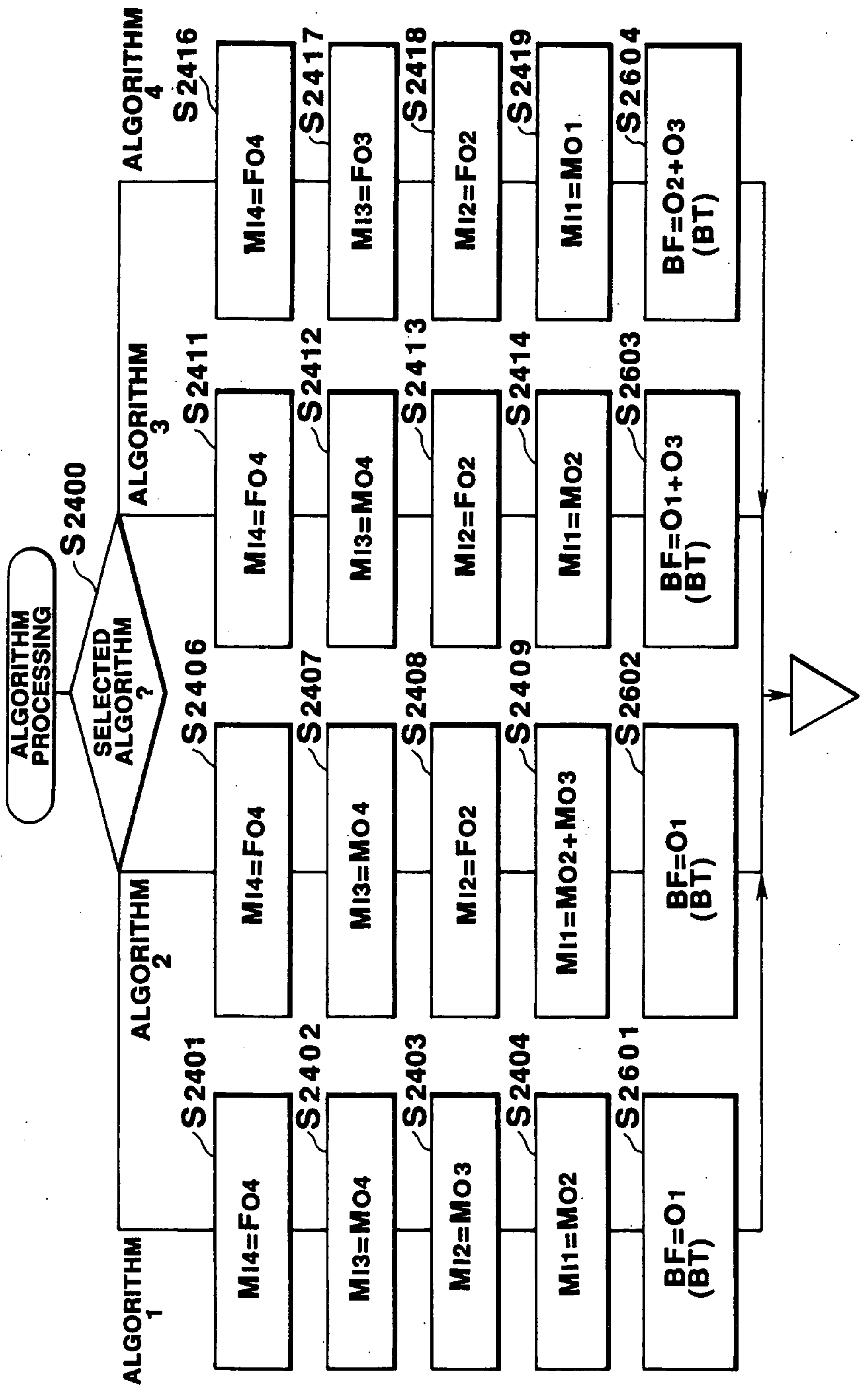

ALGORITHM PROCESSING
S2400
SELECTED ALGORITHM ?
ALGORITHM 1
ALGORITHM 2
ALGORITHM 3
ALGORITHM 4
S2401
MI4=FO4
S2402
MI3=MO4
S2403
MI2=MO3
S2404
MI1=MO2
S2601
BF=O1 (BT)
S2406
MI4=FO4
S2407
MI3=MO4
S2408
MI2=FO2
S2409
MI1=MO2+MO3
S2602
BF=O1 (BT)
S2411
MI4=FO4
S2412
MI3=MO4
S2413
MI2=FO2
S2414
MI1=MO2
S2603
BF=O1+O3 (BT)
S2416
MI4=FO4
S2417
MI3=FO3
S2418
MI2=FO2
S2419
MI1=MO1
S2604
BF=O2+O3 (BT)

FIG. 26

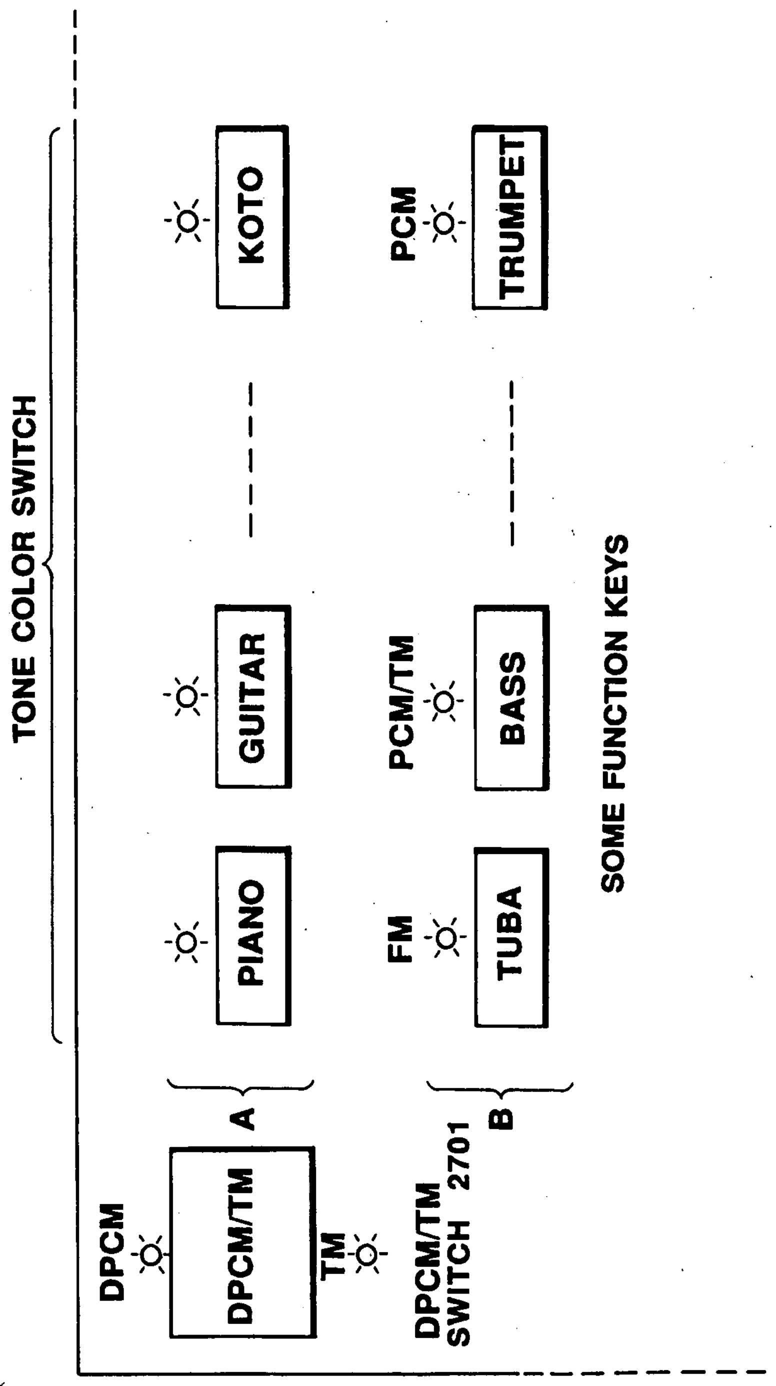

TONE COLOR SWITCH
DPCM
DPCM/TM
TM
DPCM/TM SWITCH 2701
A
B
PIANO
GUITAR
KOTO
FM
TUBA
PCM/TM
BASS
PCM
TRUMPET
SOME FUNCTION KEYS

FIG.27

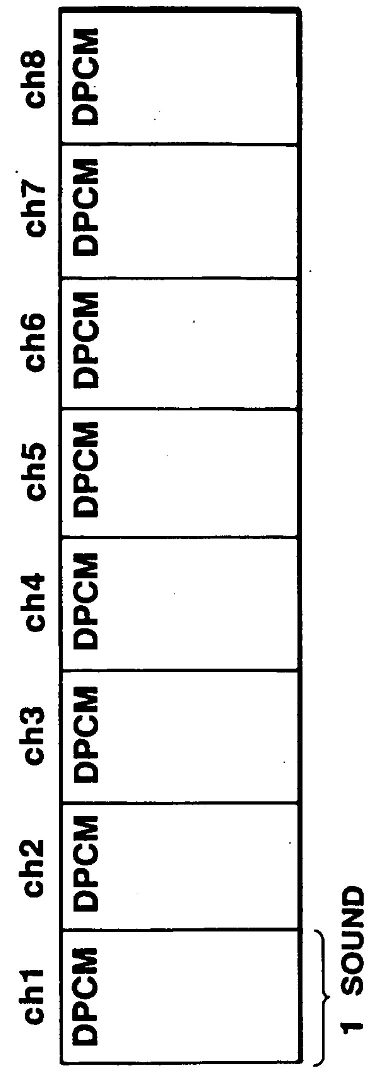


FIG.28A

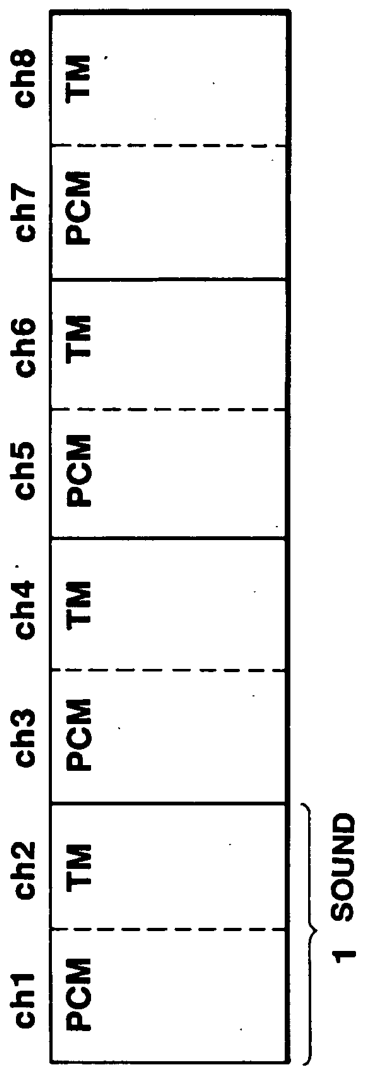


FIG.28B

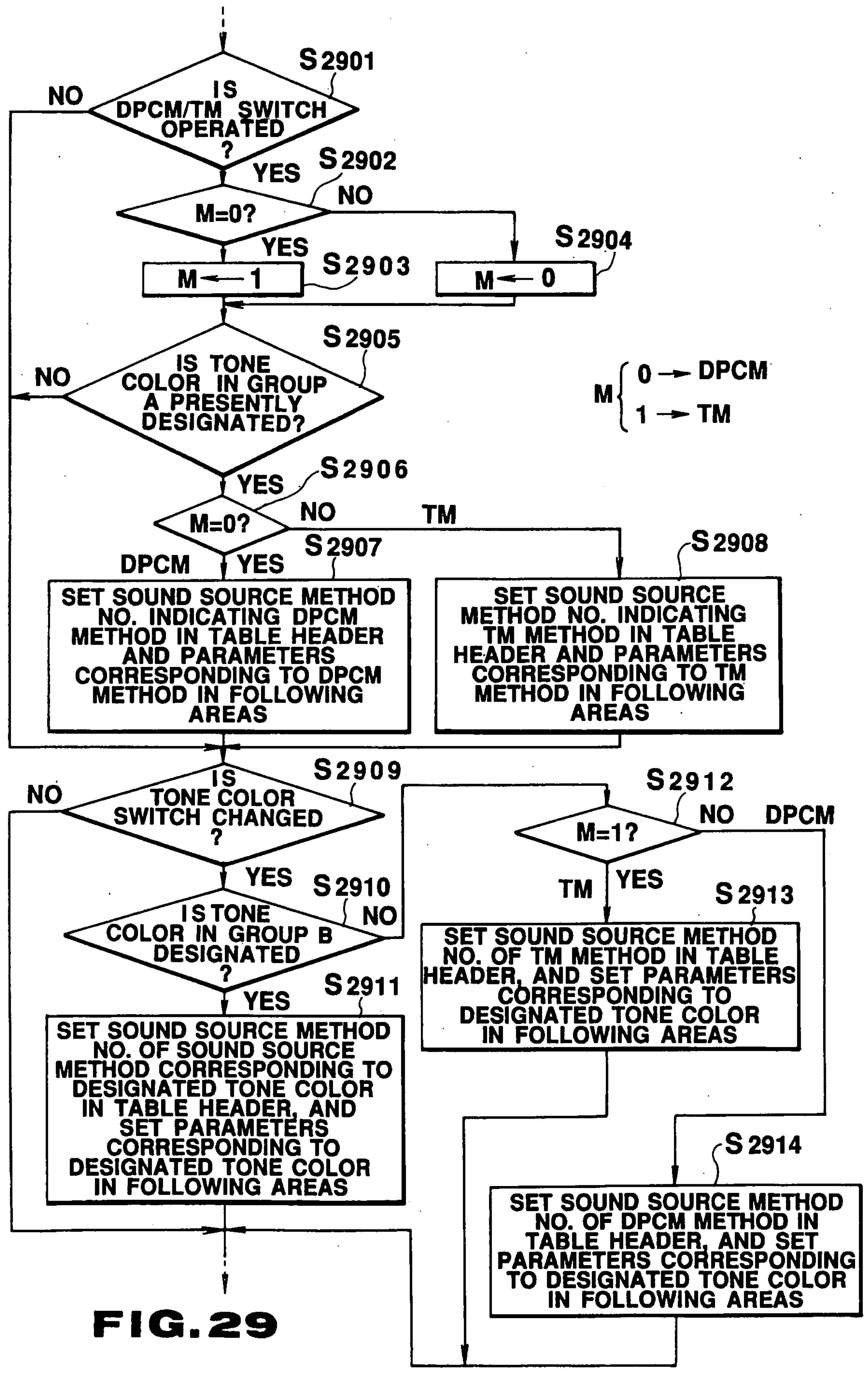

S2901
IS DPCM/TM SWITCH OPERATED ?
NO
YES
S2902
M=0?
NO
YES
S2903
M ← 1
S2904
M ← 0
S2905
IS TONE COLOR IN GROUP A PRESENTLY DESIGNATED?
NO
YES
M { 0 → DPCM
1 → TM
S2906
M=0?
NO
TM
DPCM
YES
S2907
SET SOUND SOURCE METHOD NO. INDICATING DPCM METHOD IN TABLE HEADER AND PARAMETERS CORRESPONDING TO DPCM METHOD IN FOLLOWING AREAS
S2908
SET SOUND SOURCE METHOD NO. INDICATING TM METHOD IN TABLE HEADER AND PARAMETERS CORRESPONDING TO TM METHOD IN FOLLOWING AREAS
S2909
IS TONE COLOR SWITCH CHANGED ?
NO
YES
S2910
IS TONE COLOR IN GROUP B DESIGNATED ?
NO
YES
S2911
SET SOUND SOURCE METHOD NO. OF SOUND SOURCE METHOD CORRESPONDING TO DESIGNATED TONE COLOR IN TABLE HEADER, AND SET PARAMETERS CORRESPONDING TO DESIGNATED TONE COLOR IN FOLLOWING AREAS
S2912
M=1?
NO
DPCM
TM
YES
S2913
SET SOUND SOURCE METHOD NO. OF TM METHOD IN TABLE HEADER, AND SET PARAMETERS CORRESPONDING TO DESIGNATED TONE COLOR IN FOLLOWING AREAS
S2914
SET SOUND SOURCE METHOD NO. OF DPCM METHOD IN TABLE HEADER, AND SET PARAMETERS CORRESPONDING TO DESIGNATED TONE COLOR IN FOLLOWING AREAS

FIG.29

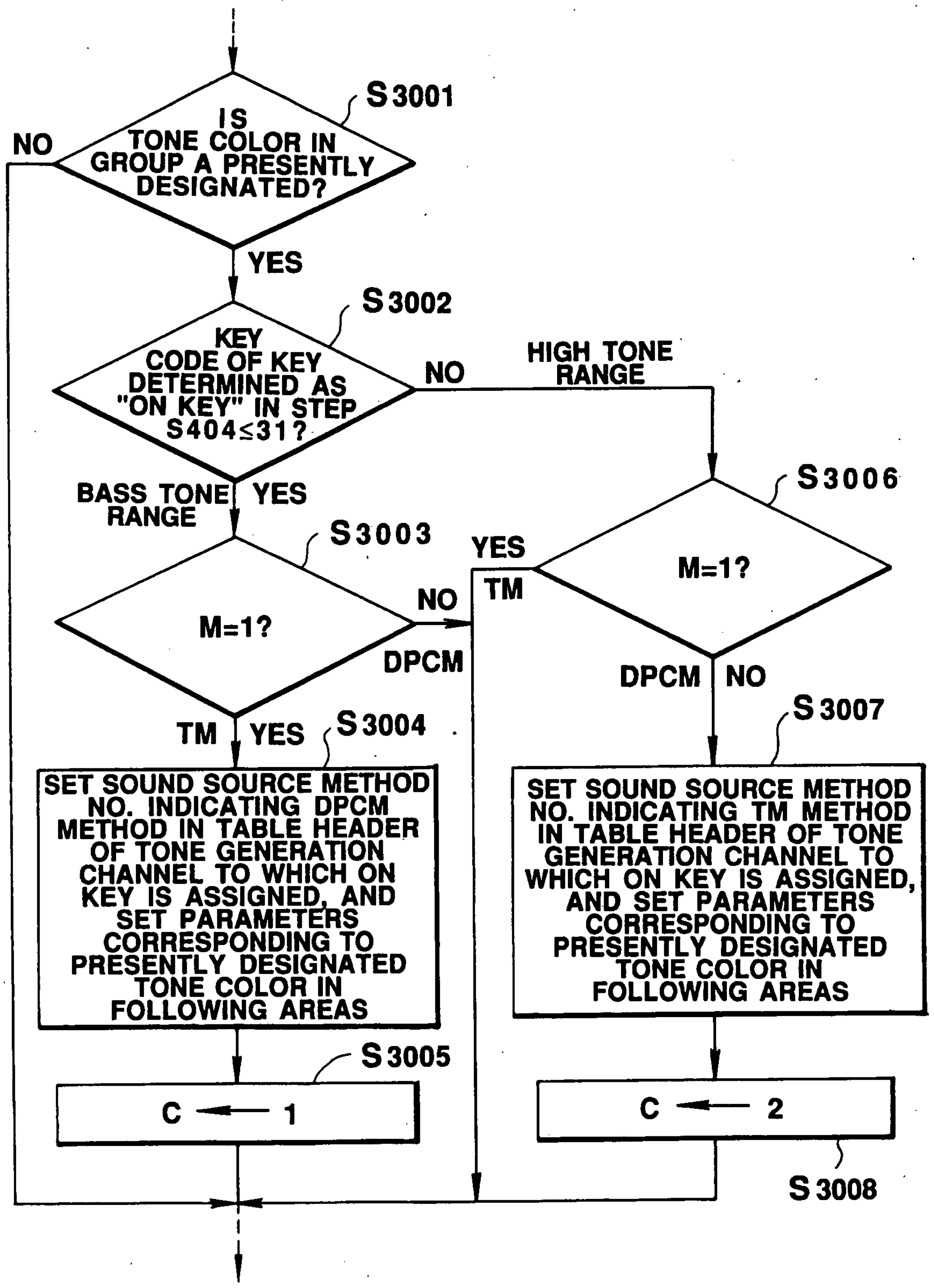

S3001
IS TONE COLOR IN GROUP A PRESENTLY DESIGNATED?
NO
YES
S3002
KEY CODE OF KEY DETERMINED AS "ON KEY" IN STEP S404≤31?
NO
HIGH TONE RANGE
BASS TONE RANGE
YES
S3003
M=1?
NO
DPCM
TM
YES
S3006
YES
TM
M=1?
DPCM
NO
S3004
SET SOUND SOURCE METHOD NO. INDICATING DPCM METHOD IN TABLE HEADER OF TONE GENERATION CHANNEL TO WHICH ON KEY IS ASSIGNED, AND SET PARAMETERS CORRESPONDING TO PRESENTLY DESIGNATED TONE COLOR IN FOLLOWING AREAS
S3007
SET SOUND SOURCE METHOD NO. INDICATING TM METHOD IN TABLE HEADER OF TONE GENERATION CHANNEL TO WHICH ON KEY IS ASSIGNED, AND SET PARAMETERS CORRESPONDING TO PRESENTLY DESIGNATED TONE COLOR IN FOLLOWING AREAS
S3005
C ← 1
C ← 2
S3008

FIG.30

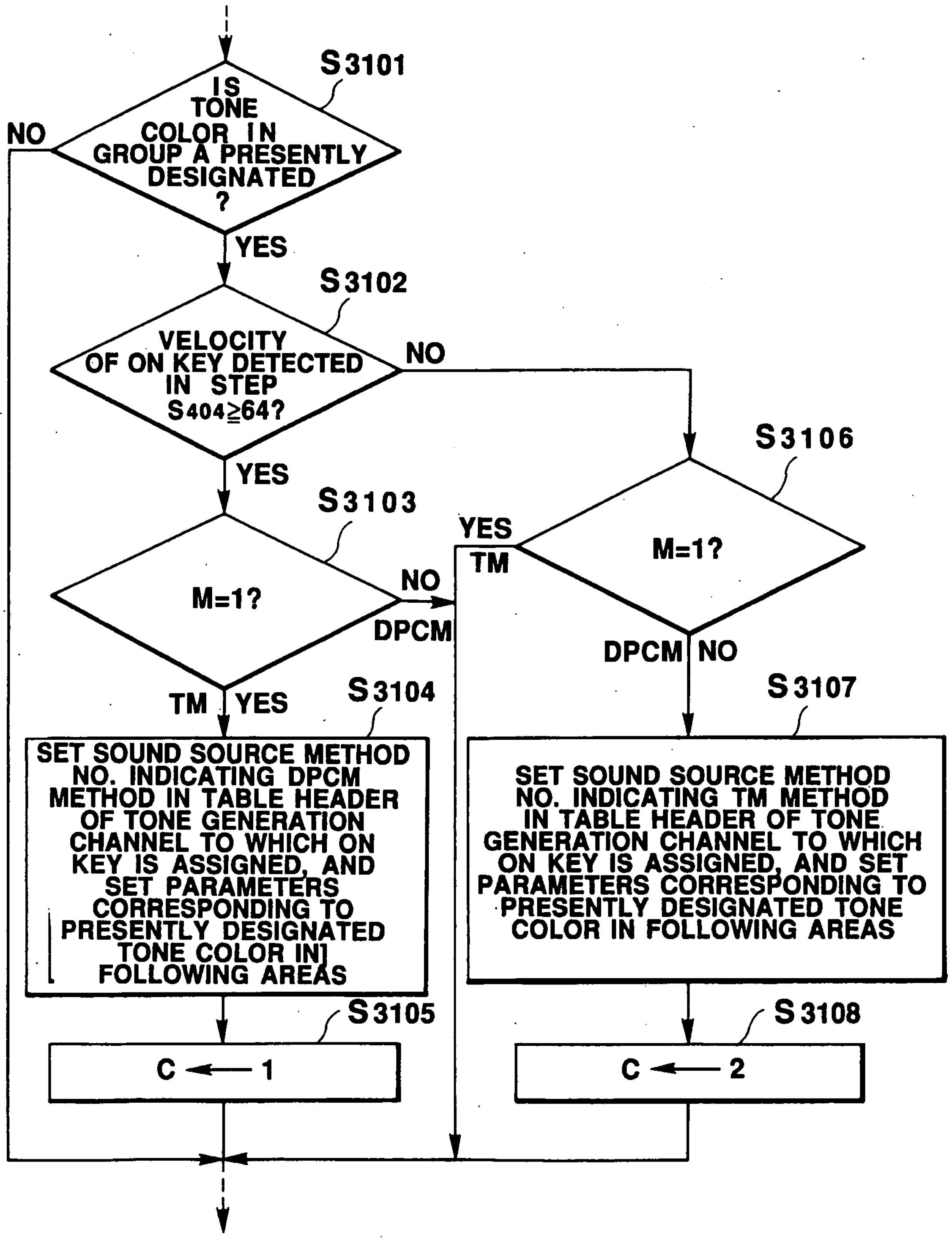

S3101
IS TONE COLOR IN GROUP A PRESENTLY DESIGNATED ?
NO
YES
S3102
VELOCITY OF ON KEY DETECTED IN STEP S404≧64?
NO
YES
S3103
M=1?
NO
DPCM
TM
YES
S3104
SET SOUND SOURCE METHOD NO. INDICATING DPCM METHOD IN TABLE HEADER OF TONE GENERATION CHANNEL TO WHICH ON KEY IS ASSIGNED, AND SET PARAMETERS CORRESPONDING TO PRESENTLY DESIGNATED TONE COLOR IN FOLLOWING AREAS
S3105
C ← 1
S3106
M=1?
YES
TM
DPCM
NO
S3107
SET SOUND SOURCE METHOD NO. INDICATING TM METHOD IN TABLE HEADER OF TONE GENERATION CHANNEL TO WHICH ON KEY IS ASSIGNED, AND SET PARAMETERS CORRESPONDING TO PRESENTLY DESIGNATED TONE COLOR IN FOLLOWING AREAS
S3108
C ← 2

FIG.31

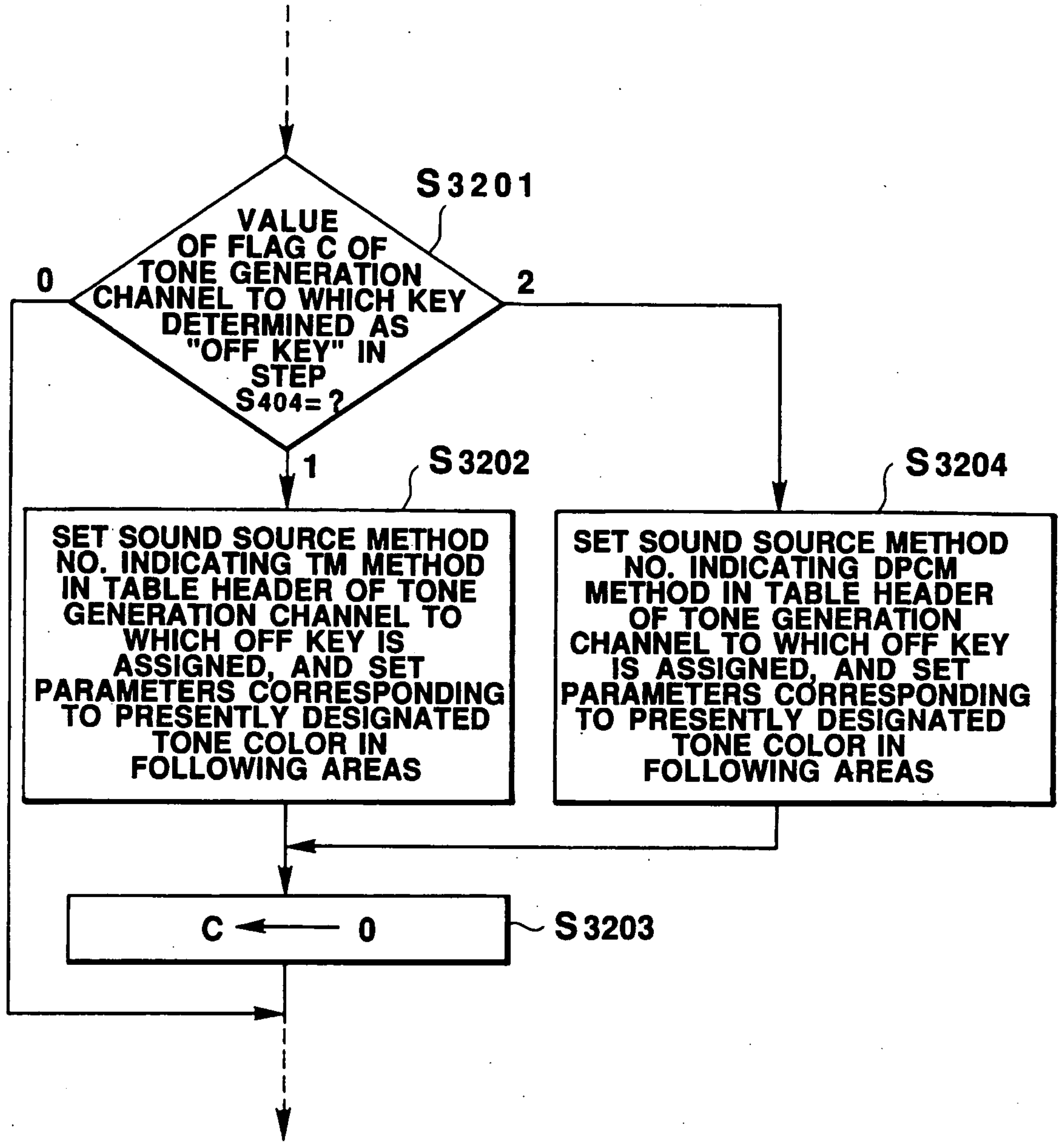

S3201
VALUE OF FLAG C OF TONE GENERATION CHANNEL TO WHICH KEY DETERMINED AS "OFF KEY" IN STEP S404= ?
0
1
2
S3202
SET SOUND SOURCE METHOD NO. INDICATING TM METHOD IN TABLE HEADER OF TONE GENERATION CHANNEL TO WHICH OFF KEY IS ASSIGNED, AND SET PARAMETERS CORRESPONDING TO PRESENTLY DESIGNATED TONE COLOR IN FOLLOWING AREAS
S3204
SET SOUND SOURCE METHOD NO. INDICATING DPCM METHOD IN TABLE HEADER OF TONE GENERATION CHANNEL TO WHICH OFF KEY IS ASSIGNED, AND SET PARAMETERS CORRESPONDING TO PRESENTLY DESIGNATED TONE COLOR IN FOLLOWING AREAS
C ← 0
S3203

FIG.32